Detection of Counterfeit Coins and Assessment of Coin Qualities

Ke Sun

A Thesis
in
the Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements for the Degree of
Master of Computer Science at Concordia University
Montreal, Quebec, Canada

July 2015

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Ke Sun

Entitled: Detection of Counterfeit Coins and Assessment of Coin Qualities

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____Chair
Dr. O. Ormandjieva

_____Examiner
Dr. T. Kasvand

_____Examiner
Dr. T. Popa

_____ Supervisor
Dr. C. Y. Suen

Approved by _____
Chair of Department or Graduate Program Director

_____
Dean of Faculty

Date _____

Abstract

Detection of Counterfeit Coins and Assessment of Coin Qualities

Ke Sun

Due to the proliferation of fake money these days, detection of counterfeit coins with high accuracy is in strong demand, yet not much research has been conducted in this field. The objective of this thesis is to introduce modern computer vision techniques and machine intelligence to differentiate real coins and fake ones with high precision, based on visual aspects.

To that end, a high-resolution scanning device – *IBIX Trax* is deployed to sample the coin images. On top of that, three visual aspects are thoroughly inspected, namely lettering, images and texture.

Six features are extracted from letterings, *i.e.* stroke width, contour smoothness, lettering height, lettering width, relative angle, and relative distance. As for classification, a hierarchical clustering – max spacing *K*-clustering—is adopted. Our experimental results show that the fake coins and real ones are totally separable based on these features.

As for images, we propose a novel shape feature— *angle-distance*. After images are segmented, a vector of size 360*1 is deployed to represent each shape. For classification, a dissimilarity measurement is used to quantize the difference between two shapes. The results show it can recognize the fake coins successfully.

As for texture, a cutting-edge feature maximum stable extremal region is adopted to automatically detect the holes and indents on the coin surface. Parameters associated with this feature are adjusted in the experiments. The detection results show this feature can be used as an indicator for assessing the qualities of coins.

Acknowledgement

I would like to take this space to have a look back of my graduate study. It has been two years of growth, fascinating experience, and most of all gratefulness.

First and foremost, I would like to give my most special thanks to my supervisor, Dr. Ching Y. Suen. During the years, Dr. Suen has given me endless guidance, supports, and encouragements. He is very open-minded and encouraging, which leaves me great space for growth. I remember the days when I shy away from opportunities, he told me to go for it. His patience and confidence in me are the biggest motivation. I feel so lucky and so grateful for all that I have been granted. I will try my best to be a person he feels proud of in years.

I'd also like to thank my parents. They inspired me to be positive, independent, and strong that till this day I still feel lucky for their education. It is the best.

To my friends, lab mates, Nicola Nobile, Guiling Guo, Boyuan Feng, Li Liu, Luyuan Li, Amir Azarnik, Juan Montes, Russ Xiang, Can Cui, Melissa Nielsen, Samantha Sieklicki, Joane Siksous, Anne Elard, they all have inspired me, and my life would be much less fun without them. I wish them all great success in the future.

Last but not the least, I am very appreciative of my examining committee's time and comments.

There were many ups and downs during the past two years, but I have learnt more than ever. And not even a day I did not feel lucky to study at Concordia University, to live in Montreal, to have the chance to communicate with people from all over the world. Thanks for the two years of experience, I feel grateful for all I've received.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Motivation

Counterfeit coins are fake coins that concern with not only valuable antique coins, but also high-value coins in circulation. They have been made by criminals for thousands of years as an illegal industry [1]. Contrary to the common knowledge that coins are rarely counterfeited because the small profits are not worth the efforts, with coins switching back to be a main currency in many countries, this is starting to change [2]. For example, in the UK, a survey undertaken by the Royal Mint in May 2014 found that 3.03% of 1£ in circulation are fake, compared to 3.04% in November 2013 [3]. Counterfeiting of coins worldwide by criminal groups or individuals has the potential to impact heavily on businesses, governments and individuals alike.

To respond, wide cooperation has been adopted by governments and banks. For example, the Australian Federal Police and the Reserve Bank of Australia have agreed on the administration of counterfeit currency [4]. Meanwhile in practice, several types of currency detectors have already been applied in self-check-out machines. The most commonly detected attributes in these devices are physical properties, which include weight, size, magnetism, metallic signature; new sophisticated coin acceptors even scan the coin and compare its image with pre-defined list [5]. However, since these machines are mainly used for high-volume transactions, the detecting processes are designed for obvious fakes. Chances are good for false negative detection results when coins are severely bent or degraded and false positive when fake coins are made with high quality.

Thus we believe a closer visual inspection is necessary. Exploration in this area of study is intriguing and meaningful for two reasons.

First and foremost, this research will help further reduce crime. Coin counterfeiting is a quick self-evolving industry. Most detection devices used in real world today are still designed for obvious fakes. Given the highly improving qualities of fake coins made nowadays, more thorough and closer inspections are necessary to detect well forged fake coins in order to provide solid evidence for detection results.

Second, this research explores a new area of application. In this thesis, we aim to introduce pattern recognition techniques to detect fake coins. Pattern recognition is a hot topic with applications varying from image to speech and text processing and such. Coin recognition is one research direction that has drawn wide interest and intensive efforts. However, as a close topic of the detection of counterfeit coins, not much research has been conducted. By means of this research, we shall apply pattern recognition techniques to recognize fake coins and fill in the research gap.

1.2 Challenge

Four main reasons are responsible for the challenge of this project.

1. ***Unusual pattern detection.*** In most pattern recognition applications, objects to be recognized are very different if they are from different classes, or similar if they belong to the same class. Thus, the objective of classification or recognition is to seek for similarities in spite of variances. On the contrary, this project focuses on differentiating similar objects, which only vary in fine and subtle details.

2. ***Size of coin.*** Part of the reason why fake coins look identical to real coins is due to their small size, usually with a diameter ranging within centimeters. By virtue of experience, experts can tell some coins to be counterfeits by touching and feeling the surface. What they are looking for in this process is the depth of lettering, sharpness of design, vagueness (fineness) of milled edge, *etc*. However, using our visual sense, fine details on both textures and image designs are almost negligible. And there is hardly any tool that can be applied directly to measure these items.
3. ***Different fake attributes of coins.*** Generally speaking, there are two major methods to make a coin: striking or casting. Though the manufacturing precision is hard to control, necessary coin-forging knowledge and skills are not difficult to learn and spread. These manufacturing techniques are unevenly distributed around the world. In some areas, coin-factories and home workshops are more epidemic than in other places. Due to varying forging techniques, there are no uniform features that can be employed to tell a fake coin from a real one. But rather, fake coins from the same source are more likely to share groups of traits, and those from different places tend not to.
4. ***Advanced forging technology and the lack of expert knowledge.*** Apart from the coins' varying qualities, maturing counterfeiting technologies are narrowing the precision gap between fake and real coins, raising the average quality level very high. In response, however, associated knowledge did not evolve much. This unbalanced development nowadays results in more hassles for the general public, and raises the need for more recognition research.

To address these issues mentioned above, or circumvent the hassles if insolvable, below lists our strategies and solutions. To note, these solutions give a general guideline for the remaining development of this thesis.

1. In response to the coin size, and potentially neglected details, we employed a novel device *IBIS TRAX* to scan and sample coin data [6, 7]. This scanning device can enlarge the original image and present fine details into high-resolution views. More information about this device is introduced in section 1.3. As a result, differences on small physical scales are exaggerated and amplified in images. In turn, features extracted from the enlarged images are more sensitive and effective.
2. Given the prior knowledge that counterfeit coins have different fake traits, in this thesis, we will extract groups of features and apply classification on each feature, whose effectiveness is tested on coin samples. A confidence level shall be given based on classification results of each feature.
3. To generate more expert knowledge, we have gathered information from various sources, to name a few, coin expert, mint factory, professional magazines, *etc*.

1.3 Machine

In response to the challenges mentioned in the last section, we employed a powerful scanning device – *IBIS TRAX* – in data sampling. Its patent belongs to *Ultra Electronics Forensic Technology Ltd. Co* in Montreal.

IBIS TRAX-3D is a scanning device that can both capture 2D image and output 3D topography of exhibits. Inside the machine, they have a built-in microscope. The scanned image results in a high resolution -- depth resolution in order of 6 microns, lateral resolution in sub-micro, which is sufficient to enable users to collect detailed data from tiny topographical peaks and valleys. Apart from that, a

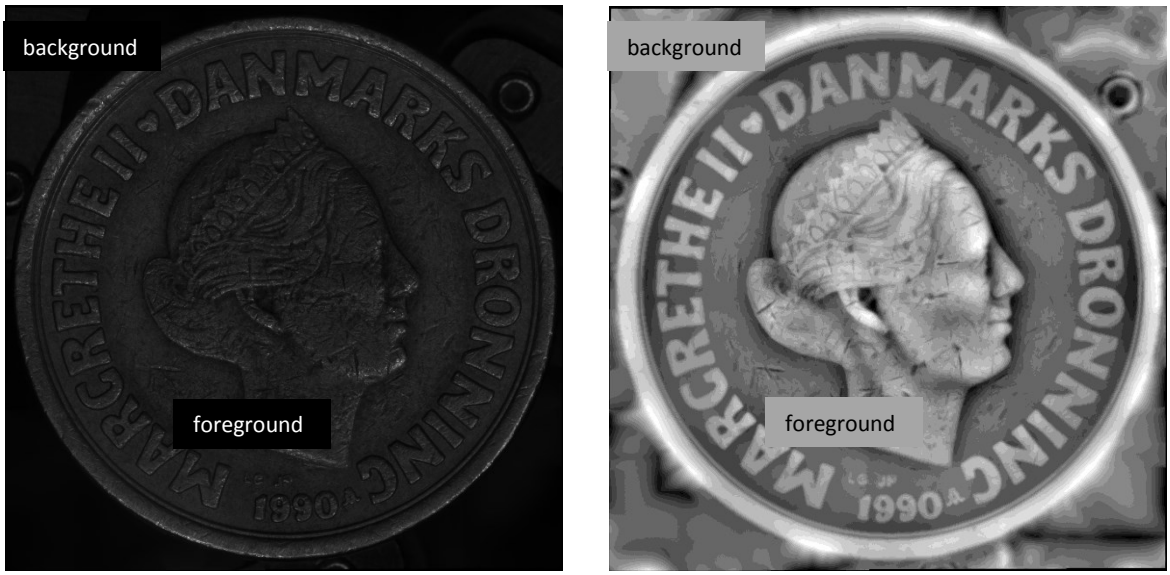
group of adjustable LEDs are configured inside too. Control of LEDs allows to view images from different angles, thus allowing dynamic observation of the surface information. By superposing the 2D texture image into the rendered 3D topography, it provides superior visualization.

Some of TRAX's main traits can be listed below.

1. 5-axis automated imaging. It provides both rotational and lateral control, and allows sufficient movement over the exhibits, thus it's able to capture accurate and undistorted image;
2. Orientation-independent annular lighting. This automatically eliminates operator variability and makes the device less prone to user error;
3. Wide field of view. This can reduce the distortion brought by stitching image patches;
4. Distortion-free orientation.

By providing a macroscopic level of details both laterally and in depth, IBIS TRAX-3D will enable us to better view and analyze the fine details on the coin surface.

Fig. 1-1 shows a typical coin image sampled from *IBIS TRAX-3D*. Fig. 1-1a is a 2D grayscale image that contains texture information; Fig. 1-1b is the corresponding depth image, meaning coin depth information is coded and represented in intensity value. Generally speaking, brighter areas have higher intensity values, and accordingly, correspond to the raised parts on a coin surface; while darker areas are sampled from more flat surface, indents, holes and such.



a. 2D grayscale image **b.** Depth image

Fig. 1-1 Image example from *IBIS-TRAX-3D* (Source: *90_Dr Suen_01*)

For any pair of coin images, its grayscale image and depth image correspond exactly. And thus, manipulations on one image are transferable to the other.

In this thesis, we will explore methods of using both 3D topographical depth image and 2D texture image, trying to solve the problem from a combination of different perspectives by fully taking advantage of both types of images.

To clarify, the centroid of coin does not overlap with the center of the rectangle image frame. And between different images, there is a tiny difference on the relative position of coins to their frames.

1.4 Literature Review

Detecting counterfeit coins using machine intelligence has not attracted much attention so far. However, major advances have occurred in a close topic, coin recognition. These two topics share a lot of similarities. For example, how to prepare and process images, what features are employed, and how they are extracted and used in classification. Studies on coin recognition have given us

inspirations regarding those topics. Thus in this section, a broad overview on coin recognition is briefly summarized. Counterfeit coin detection is discussed in [8].

During the past two to three decades, much research has been conducted in the field of visually based coin recognition. Based on some studies, several automatic recognition systems have then been invented and put into practice in industry: Dagobert, from Austrian Research Centre (ARC), Seibersdorf [11], is able to recognize more than 600 coin types with more than 2,000 coin faces; COIN-O-MATIC, from MICC-IKAT [17, 19], can classify approximately 72% of the coins correctly, while misclassifying only 2% of the coins. A mature recognition system always consists of 4 key components: image capturing, pre-processing, feature extraction and classification. Some studies have also included a verification phase [12, 13, 19].

Image Processing

Intuitively, while humans are trying to recognize coins, their visual perspectives usually cover the following three aspects:

1. ***Image:*** image is the stamp pattern on coins surface; common images are for example, celebrity profile, animal, emblem, *etc*.
2. ***Texture***: texture is also known as surface topology, a feature reflects roughness, uniformity, brightness *etc*. of the surface;
3. ***Lettering***: letterings include letters, characters, digits or other meaningful symbols on the coin surface.

Lettering provides straightforward information for manual recognition. However, it is not quite desirable in terms of implementation, given the computational complexity and immaturity of text recognition techniques. Thus most studies fall

into either image-based recognition [9, 14, 16], or texture-based recognition [11, 12], or as in some studies, a combination of both [17, 19].

Whichever perspective is inspected, to well prepare images for feature extraction, a successful system needs to address 3 problems, namely illumination invariance, translation invariance, and scale invariance.

Illumination variance results from varying coin surface reflection due to abrasion, dirtiness, or uncontrolled lighting. Due to heavy circulation and poor quality, some images suffer from low intensity, poor contrast, and vagueness and such. Though usually subtle, illumination variance can introduce noise to feature extraction and classification. Problems of how low intensity can affect recognition results are reported [17], and desires on improvement of contrast stretching are elaborated [19]. In order to circumvent this problem, *Reisert, M. et al*. [9, 16] used gradient images. *Huber* [13] has conducted a more thorough research: performances of four eigenspace approaches are compared in terms of reduction of illumination impacts.

Translation invariance are normally solved by coin segmentation [9, 11, 16], also called coin detection [9]. Popular methods for segmentation include: Hough Transform, generalized Hough Transform (GHT) [9, 12, 16], where focus of area is a circular coin; or Ant Colony optimization [11], where selected area can be of any shape. Once coins are separated, with their centroids located, radius found, coin images are normalized to be scale invariant. Normalization parameter can also be used as accessory features [13].

As for texture-based analysis, the most common processing approach is spectral transformation, *i.e.* Wavelet [10, 11, 12]. By decomposing the original image into sub-images of desired frequencies and directions, wavelet has demonstrated outstanding performance in texture segmentation and such. The advantages of

introducing wavelet are two: *first, lower image resolution* – a lower resolution means lighter computation, less redundancy; second, *extract high-level texture feature*. There are slight differences at the implementation phase in individual works though, with regard to what channels to decompose, how many levels to choose from, and so on.

Feature Extraction

The most influential and tricky problem in feature extraction goes to rotational invariance. Under all circumstances, coins' pointing directions are not fixed, and are difficult to be aligned while being sampled. To solve it, different methods are explored. Common techniques include but not limited to: ring-projection transformation, usage of statistical features, and extraction of rotation-invariant features.

In [9, 16], quantized radial gradient directions are calculated and used as features for further alignment. Theory that proves its rotation-invariance can be found in [20]. In [10], the reserved sub-image is divided into concentric rings and the mean energy value for each ring is extracted as features. Similarly, in [12], mean and standard deviation of K rings in all wavelet images are concatenated; besides it also selected Local Binary Pattern (LBP) images as features. In [14], another rotation-invariant feature – edge points, obtained by applying Canny edge operation and the Laplacian of Gaussian – is chosen. Likewise, authors in [17, 19] also used edge images, which are convolved with two orthogonal Sobel kernels; then distributions in circular rings (called Edge-Distance feature) and in sectors (called *Angle-Distance* feature) are measured as features.

Classification/ Registration

In order to label a testing coin, two solutions are most explored: image registration [9, 16], and feature classification [10, 12, 13, 15, 17, 18, 19]. Some paper used a combination of both [14].

Image registration, also known as image alignment, is to adjust the pose of the current image to best align with the reference image till the differences in translation, scale, and rotation are minimized. Usually a distance thresholding is applied after alignment. The idea is that in ideal cases where images are much alike, distance should be as small as within tolerance. Given that coin images are translational and scale invariant, precise rotational alignment can be implemented using full resolution grayscale image. However, this is almost impossible considering the data size. Researchers would take advantage of features that they extracted from the previous stages. For example, alignment in [9, 16] is implemented using gradient, in an exhaustive fashion, where the testing sample is compared with all the training data in all subtly divided directions. Master image that gives the minimal difference should be the match of the testing coin, and the direction which gives the minimal dissimilarity is the best alignment. Though it took advantage of Fourier transform to speed up calculation, the computation burden is still quite considerable.

Image classification is to classify images into different categories based on features. The most common classifiers found in the literature include: K-Nearest Neighbor (KNN) [10, 12, 17, 19], Neural Network (NN) [15], Bayesian fusion [13], *etc*. Based on different distance formulas embedded in KNN, the methods can further be subdivided into registration and correlation. Note that the choice of

classifier is closely related to the features extracted and the specific processing methods of raw images.

Verification

Verification is necessary under two circumstances: (a) The testing dataset includes unknown coins; (b) False positive and true negative recognition are very expensive. *Shen et al.* used a distance threshold [12], where if the closest sample of the testing coin from training set is beyond a certain threshold, it is discarded as unregistered; *Huber et al.* [13] adopted a rejection threshold based on posteriori probability, while *Van der Maaten et al.* [19] used a mutual information threshold based on blurred intensity gradient.

Result Analysis

The system in [9] shows a very good classification result. When no verification is applied, the result is even slightly better (compare Tranche 1A (no reg.) and 1A in Table 1 [9]). The correct recognition rate can reach approximately 97% and the false positive rate is very small. It proves that direction information is a reliable feature and robust to change of illumination and contrast. It is also stated that angular/radial resolution of 256*64 is ideal. *Shen et. al.* [12] use a training dataset of 8,762 coin images and a test set of 2,200 images. The result suggests that rotation invariance feature CSGabor together with *Nc* distance outperforms Gabor histogram, LBP histogram, SOGabor and so on, also is better than multi-scale edge distance histogram. *Van der Maaten et. al.* show that edge distance histogram is a much better feature than angle histogram, yet the highest recognition ratio is less than 70% [17]. *Nolle et. al.* also give a very promising result: 95.79% of 214 known coins are correctly classified and 99.7% of 334 unknown coins are

successfully rejected [14]. Yet with a large testing set – 12949 coin images—ratio of incorrect rejection can be as high as 15.11%.

Other state-of-art methods like Neural Network are also deployed, yet they have not shown outperforming recognition results [15, 18]. For a more overarching review, readers can refer to [21].

1.5 Thesis Outline

In this thesis, we shall check three visual aspects on the coin images: they are respectively letterings, images and texture. Thus the main content can be briefly summarized into 3 parts: (a) letterings separation from image background and features extracted from letters and digits; (b) image separation and a novel shape feature; and (c) application of a distinct region (DR) feature maximally stable extremal region (MSER) for texture analysis. Below describes the detailed organization for the following chapters.

Chapter 2 will first talk about some image processing techniques, including Hough transform, global and local thresholding, blob extraction and so on. The purpose of this phase is to isolate each of the letters and digits such that they can be individually used for feature study. Once lettering segmentation is ready, four intra-letter features are extracted, including letters' stroke width, smoothness, height and width; and two inter-letter features are studied, which are relative distance and relative angle between adjacent letters. Then we shall test two groups of coins and show the experimental results to verify the effectiveness of those features.

Chapter 3 will introduce a novel shape feature, "*angle-distance*". To get this feature, two methods of image separation are introduced, namely *contour tracking* and *foreground separation*. Then we will show how to get the *angle-distance*

vector from the isolated image, and also prove why this is valid and computationally efficient. Last but not least, we will present the experimental results and demonstrate the validity of this feature.

Chapter 4 will focus on texture analysis. In the beginning, a brief review over texture analysis is provided, and from all the solutions we chose an affine invariant feature, MSER. Next, we will talk about how MSER is applied to detect holes and indents on coins. In the section of experiment, we will discuss the optimal parameters of MSER, and how the adjustment of those inputs can affect our results.

Chapter 5 will draw conclusions on our work so far. We will talk about the main contributions of this thesis. Also more future research directions will be presented.

Chapter 2 Lettering Analysis

If a coin is struck using a fake die as most often is the case, by and large, letterings on fake coin will be different. The purpose of this chapter is to analyze the letterings on the coin face, by revealing and checking potential fake traits. This chapter is organized as follows: first in section 2.1, necessary image processing steps are carried out to prepare separated letters. Methods of coin detection and segmentation are discussed in this section, and different approaches to separate letters are elaborated and compared. In section 2.2, based on the extracted letters and digits, six features are explored, including four intra-lettering features, *i.e.* stroke width, contour smoothness, letter height and width; and two inter-lettering features, *i.e.* relative angle and relative distance. Section 2.3 talks about a hierarchical clustering method, max spacing K-clustering. It is an unsupervised clustering method based on Euclidean distance and the model is evolved from minimum spanning tree. In the last section, we test two groups of coins and by checking the consistency and discrepancy of each feature, we proved their effectiveness.

2.1 Image Processing

2.1.1 Segmentation

In practice, an image is usually separated into background and foreground. In this application, background is the dark area outside coins rim, as shown in Fig. 1-1b. The background does not convey useful information, yet it consumes RAM and lowers calculation speed. Apart from that, as acknowledged in Section 1.3, there exists nuance of relative position of foreground coin image to background frame, which is responsible for translational variance.

In order to get rid of possible negative impacts from the background and align coin images to be translational invariant, the coins are detected and segmented from the background. The most commonly used segmentation method is Hough transform [22]. It is a commonly used technique in digital image applications. The purpose is to detect and locate shapes by voting, as long as the shapes have a parametric presentation. This voting procedure is carried out using a so-called accumulator space, which is a subdivision of parameter space.

Hough Transform

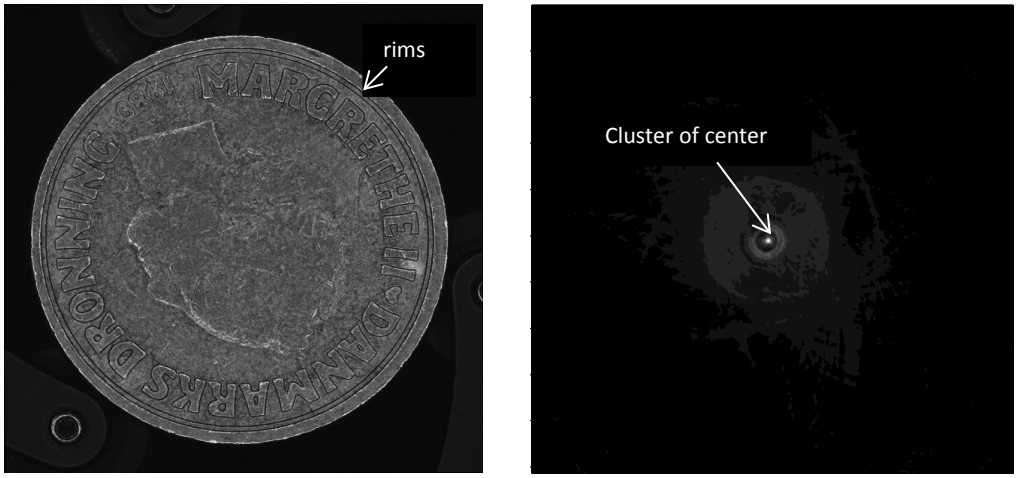
Below are the detailed steps of Hough Transform.

1. *Obtain a binary edge image*: the purpose of using binary edge image instead of grayscale image is to reduce computational workload;
2. *Specify the sub-divisions in x-y-r-plane*: create an accumulator space that is made up of cells. Each cell stands for one pixel. Initially setting all cells to 0;
3. *Examine the counts of accumulator cells for high pixel concentration*: for each edge pixel, find its center candidates, and increase by 1 at the cells corresponding to all candidate pixels;
4. *Search for the local maxima cells*: these are cells with highest votes.

In practice, because computation grows exponentially with parameter space, it is ideal to keep the number of parameters small [23]. In this application, due to the prior knowledge that a circle exists and its radius does not change, we narrow the *x-y-r-plane* to *x-y-plane*.

Fig. 2-1 illustrates an example of Hough Detection. Fig. 2-1b is the visualized Hough accumulator. The local maxima is not obvious though, which makes sense: check the coin image in Fig. 2-1a, there exist 3 rims and they are very close to each

other. In turn, the voting space around the center area is enlarged and vague. However, since a standard circle is isotropic, and its edge point has equal voting, the enlarged cluster of center is isotropic too. Thus the strategy is to filter the accumulator with a threshold to only leave the most voted area, whose center is taken as the coin centroid.

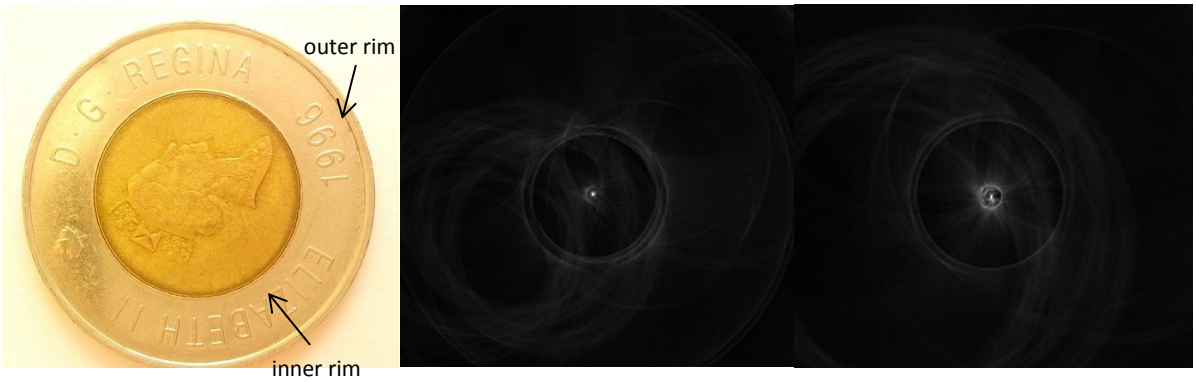


a. Testing image **b.** Hough accumulator

Fig. 2-1 Hough transform

Apart from segmentation, Hough transform is also used to pre-check if coins have standard rims. As soon as coin centroid and radius are known, its rim can be plotted exactly. Compare the rim and the image. If they do not overlap totally, the coin shape is not exactly round, or there is protrusion or indent on the rim. Thus it can be easily detected and rejected as a fake coin.

Applications of rim checking can be extended further. For coins with more than one rim, for instance, Canadian *Toonie*, segmentation can be applied twice to find the two centroids, corresponding to the outer and the inner rim respectively, as shown in Fig. 2-2. If the distance between the two centers is above a certain value, it is classified as a fake one; otherwise it will be forwarded for checks of other aspects.

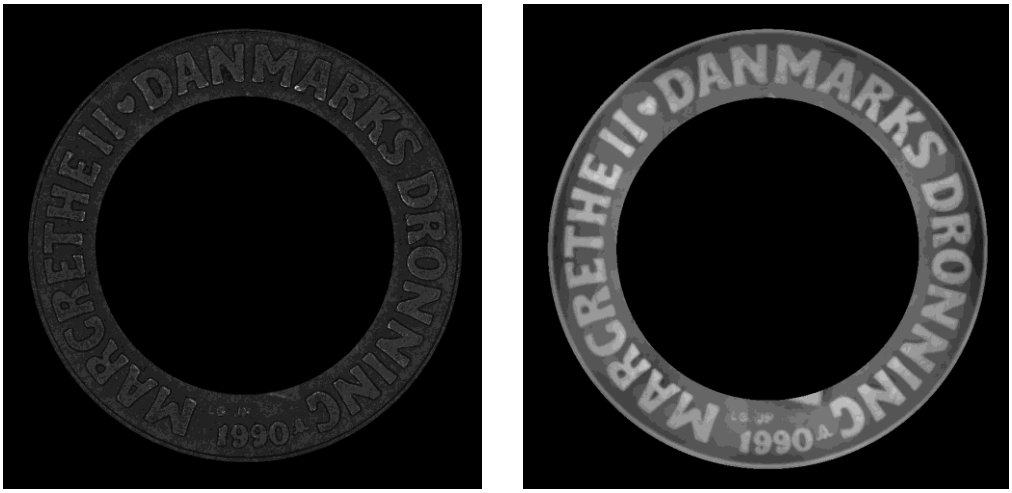


a. Testing image **b.** Inner rim accumulator **c.** Outer rim accumulator

Fig. 2-2 Double rim application

2.1.2 Binarization

After coin segmentation, the centroid of the coin in the image is precisely located. As for the coins in this research, all the letters and digits are distributed within a ring centered at coin centroid. To separate letterings, it is more efficient to narrow the coin image into the ring size, which includes only the letterings, as shown in Fig. 2-3. This can not only lower computation since a large amount of image is cut out, but also reduce noises from irrelevant pixels for future operations.



a. Grayscale image lettering ring **b.** Depth image lettering ring

Fig. 2-3 Letterings separation

Theoretically, there are two ways to isolate each of the letterings. They both have strengths and weaknesses.

First, edge detection. It's a straightforward idea: edges make up the contour of letters and digits. Thus if edges are extractable, the focus will be the area inside the edges. The limit is, however, most often edges extracted are broken. Fig. 2-4a shows the result of Fig. 2-3 after Canny detection. Hardly is any edge connected. A backup is to use thresholding approach. As noticed in grayscale images, edge points are of high contrast with their surroundings. A global thresholding method like *Otsu* [24] should be able to detect edges. Fig. 2-4b shows the thresholding result after de-noising. The effect is better, however, there still exist broken points, and this method brings in more noise.

a. Separate letterings by Canny **b.** Separate letterings by Otsu

Fig. 2-4 Separate letterings by edge detection

Second, binarization. Instead of grayscale images, depth images are used in binarization. Note that there is a very sharp intensity contrast between letters and their surroundings. It is a very stable trait since letterings are raised above the surface. Unless a coin is totally worn out to be flat, the contrast remains to be

obvious. Our approach is to use Niblack [25], a local thresholding method, to separate the letterings. Fig. 2-5a shows the result after thresholding. By comparison, binarization result is clearer, more continuous and has less noise than the results from edge detection.

Fig. 2-5 Separate letterings by binarization

Since the second method performs fundamentally better, it is adopted in this research. However, notice in Fig. 2-5 that though letterings are separated, there are undesirable noises surrounding the letterings. A few follow-ups are necessary to get rid of irrelevant areas, meanwhile to locate each letter and digit.

To that end, we used the connected component labeling [26]. Connected component labeling is also called blob extraction/detection, or region labeling. A connected component is defined as a cluster where all pixels share the same intensity, and are connected in some way, for instance, 4-neighbor-connectivity, 8-neighbor-connectivity and such.

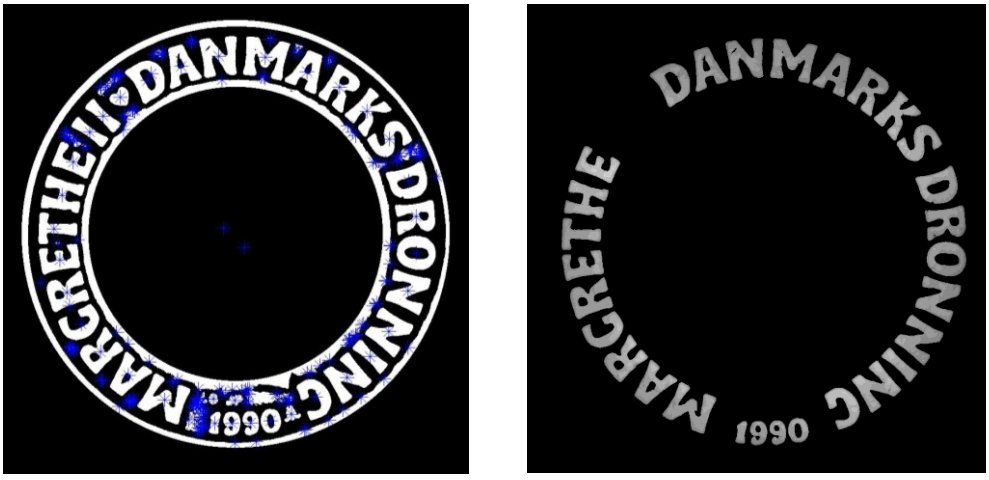
Blob Extraction

As the input is a binary image with the focus of interest valued 1, and background 0, the idea is to traverse all pixels by moving row by row. When it encounters a pixel *p* whose value is 1, check *p*'s neighbors. The 3 situations to happen are listed below:

1. All neighbors of *p* are 0, then assign a new label to *p;*
2. Only one of *p*'s neighbors is 1, then assign its label to *p;*
3. More than one neighbor with a value of 1, then assign any one of its neighbors' labels to *p,* meanwhile, mark the equivalence of its neighbors' labels*;*

Finally, after all pixels have been traversed, merge the marked labels.

An example from a small patch is shown in Fig. 2-6: points in blue are centers of the blobs. By tracking the labels, letterings can be segmented in Fig. 2-6b. As acknowledged from Section 1.3, because the grayscale image and the depth image have exact correspondences in position, letterings from grayscale image are also extractable.



a. Blob extraction result **b.** Locating letterings

Fig. 2-6 Lettering segmentation results

2.2 Feature Extraction

In this section, the focus is to extract tentative features based on previously separated letters. Four independent lettering features adopted in this thesis are stroke width, outline smoothness, height and width; besides, two inter-lettering features, relative distances and angles, have also been explored.

2.2.1 Stroke Width

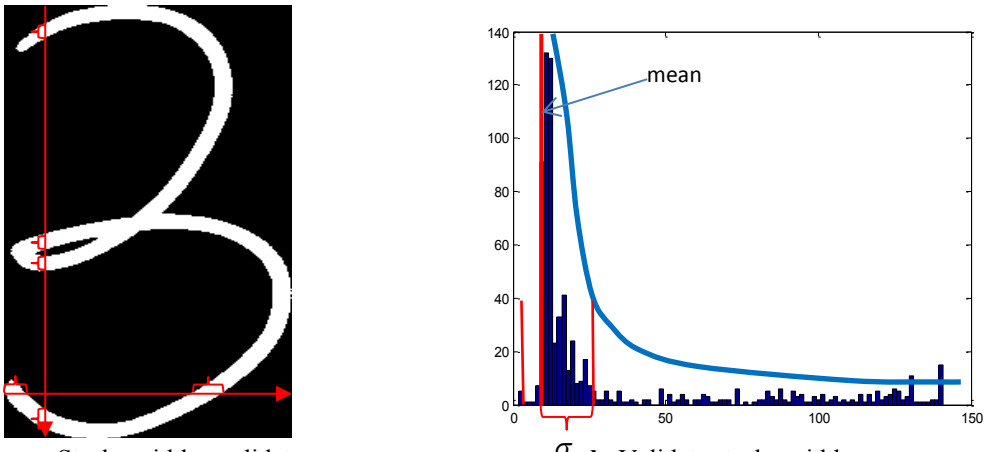
Stroke width is the thickness of the stroke of a letter or digit; this attribute specifies the width of the outline.

Given a binarized text image, many methods have been developed to calculate the stroke width. In 2009, for example, *Ntirogiannis et al*. [27] proposed an adaptive detection method. First, binarized letterings are skeletonized. For each skeleton point, its nearest point on contour is found. Take the distance as *D*, assign *2*D+1* to that skeleton point, which is recorded as one associated stroke width. If a letter or digit has more than one connected component, then in turn, it has more than one skeleton. For each skeleton, they took the maximum stroke width. Average stroke width of all skeletons belonging to one letter or digit is taken as its final stroke width. This method works well when a stroke has even width, which is not very true in most applications. Apart from that, skeletonizing letterings and finding nearest points for all skeleton points can be very time consuming.

In this thesis we propose a new method to calculate stroke width. Given the fact that a stroke is not evenly wide at different parts, a statistical method is applied to best respond to the width distribution. Two accessory concepts need to be explained. *First, valid stroke* – at some points, when either the stroke width is too small or too large, they are regarded as invalid and are discarded. Validation

process is demonstrated in implementation steps shown below. *Second, average value* – the mean value and the standard deviation of the valid stroke width candidates are calculated, and back to part one, values that are beyond three times away from the mean are further discarded. Note that this is a heuristic method that the results are close enough to real stroke width. Below are the detailed steps.

1. Take contour image as input; initialize an empty set *swAll* { };
2. Traverse the image column by column, left to right, and put in the set *swAll* all valid stroke width (SW) candidates, shown as red left brackets in Fig. 2-7a; repeat this row by tow, top to bottom;
3. Use a normal distribution to fit data in *swAll*. Calculate the mean μ and standard deviation σ of this set; delete all stroke values outside the range of $(\mu \pm \sigma)$, as shown in Fig. 2-7b;
4. Calculate new mean μ' and standard deviation σ' after Step 3;
5. Check if the mean value converges: if $|\mu - \mu'| < 0.2$, set SW as μ'; otherwise go to step 3.



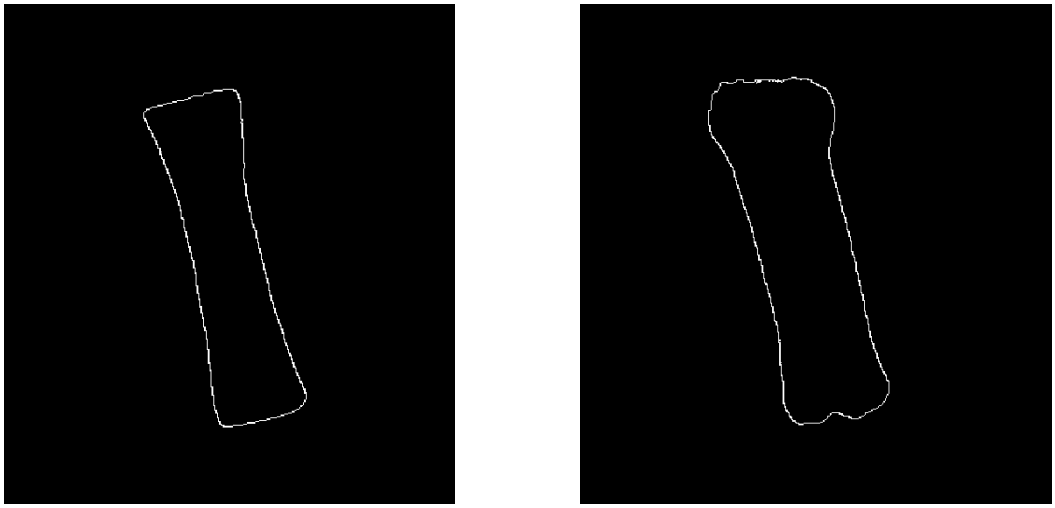
a. Stroke width candidate **b.** Validate stroke width

Fig. 2-7 Detection of stroke width

The example shown in Fig. 2-7a, digit '3' has an evenly distributed stroke width of 12 pixels. Calculated result is 12.3268, proved it is valid. Further examinations are carried out in section 2.4.

2.2.2 Outline Smoothness

Fig. 2-8 shows a pair of letter contours**.** Fig. 2-8a is extracted from a genuine coin: the outline is smooth, free of indents or protrusions along the contour, whereas for the letter from a counterfeit coin as shown in Fig. 2-8b, it consists of unexpected bumps, curves, and in general is very coarse. This trait is very common for low quality fake coins, which makes sense: making sharp and neat edges requires very costly machines and striking/casting dies. Either is practical for most low-cost counterfeit coins. In turn, when the images are enlarged, flaws of coarse contour become obvious.



a. Contour of genuine coin (Source:*96_ref_b*) **b.** Contour of fake coin (Source: *96_KT3*)

Fig. 2-8 Contour comparison

For two shapes whose contours are slightly different, *i.e.* their shapes resemble but one has unexpected bumps and indents along its contour, their enclosed areas are not as much affected as their contour lengths. From there, we adopted ratio of

enclosed area to contour length as a feature to represent contour smoothness. The enclosed area equals to the number of pixels inside the contour, while contour length is the number of pixels along the contour.

For the examples shown in Fig. 2-8, their ratios are 21.36 and 27.52 respectively. The latter is 22.4% larger than the former. More testing and analysis will be described in section 2.4.

2.2.3 Principal Axis

Principal axis is synonymous to the main direction, principal orientation, *etc*. Basically, it represents the pointing direction of a shape, and reveals information on how a shape spreads. Several methods have been proposed with regard to the principal axis, which can be categorized into 2 classes.

First, principal component analysis (PCA). PCA is a statistical procedure that concerns with converting standard Cartesian coordinates where observations are possibly correlated, into new coordinates, where data if transformed into, their co-variance between axes in new coordinates is 0. New axes are called *Principal Components*. They are perpendicular to each other, and a weight is assigned to each component. The axis with largest weight is what we are looking for as the principal axis. This method allows researchers to identity the principal directions in which data varies. However, PCA is mainly applied to discrete points.

Second, moment theory. Moments are a set of specific quantitative measurements, of the shape of a set of points [28]. It describes image distribution with respect to axes. The 2-D moment of a shape $f(x, y)$ in the order $(p + q)$ on image of size *M*N* is defined by Eq. 2-1.

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q f(x,y) \quad 2\text{-}1$$

Here p and q are integers. Eq. 2-2 defines the corresponding central moment.

$$\mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x-\bar{x})^p (y-\bar{y})^q f(x,y) \quad 2\text{-}2$$

Obviously, the central moment is independent of positions.

Moments of different orders have different focuses on shape distribution.

- The zeroth moment: total mass or the area of shape
- The first moment divided by the zeroth moment: center of mass
- The second moment: variance
- The third moment: skewness of distribution

Principal axis based on moment theory is described as this: if a rotational angle θ, which makes the second order central moment μ_{11} minimum, it is the angle of new principal axes with reference to *x, y* axis. This θ can be obtained from Eq. 2-3.

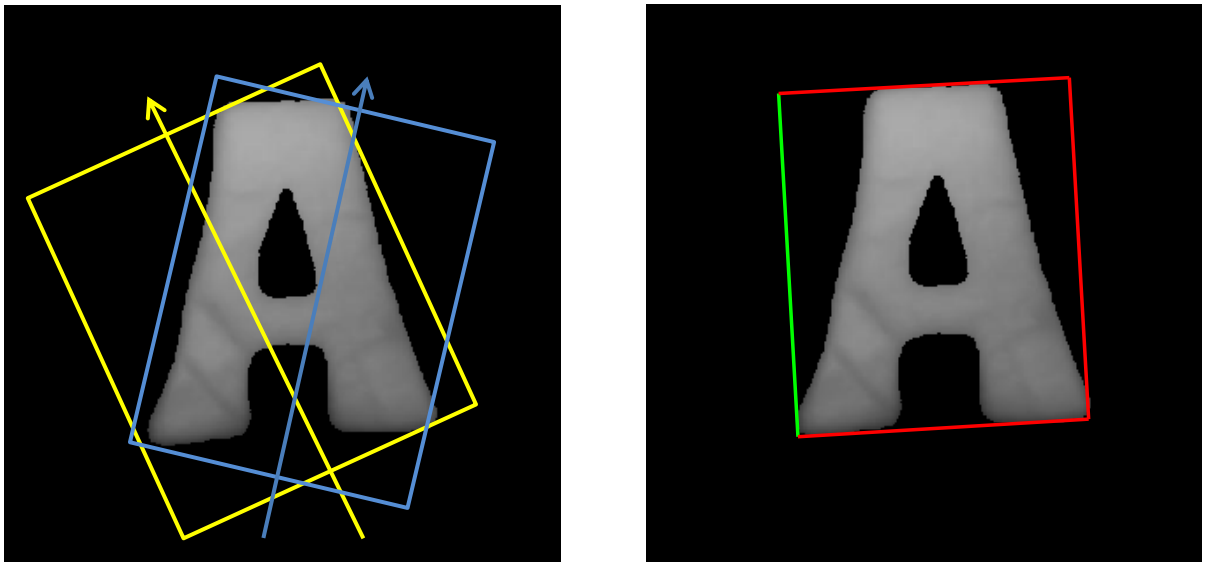
$$tan2\theta = \frac{2\mu_{11}}{\mu_{20}-\mu_{02}} \quad 2\text{-}3$$

We tested examples using moment theory. However, while some principal axes are very close to the ground true direction, other results turned out to be unreliable. A pattern we found is that, for letters like A, M, *etc.* which are left-right reflectionally symmetric, results are the most reliable; letters like N, which is rotationally symmetric, results are the least reliable; reliabilities of the rest letters spread in the middle of this spectrum.

In this thesis, instead of using moment theory as the final results, we use it to give a rough approximation. Based on that, we proposed to add another constraint to find the principal axis. To note, a very important premise of this new method is that

letters and digits must stand upright on its bottom. They should not be slanted as most often occurs to scrawl handwriting. In this application, all the letters meet this condition.

The idea is to superimpose a rectangle box of a random direction onto the letter. Shrink its size till it just fully covers the whole letter. This box is called the bounding box of that direction. As depicted in Fig. 2-9a, arrows pointing arbitrarily stand for directions. The rectangle in the same color is the bounding box associated with that direction. When rotated, the box will change, with one border lengthened, and another shortened, and its size might enlarge or shrink.



a. Bounding box of arbitrary direction **b.** The minimum bounding box

Fig. 2-9 Principal axis

The size of bounding box will reach minima only when the principal direction of it coincides with direction of the letter, in which case it is called minimum bounding box, or enclosing box. Its two principal directions are respectively principal axis and minor axis of the letter. As depicted in Fig. 2-9b, the green boarder shows the principal axis, and the frame is the minimum bounding box.

Concrete steps to extract the principal axis are described as below:

1. Use moment theory in Eq. 2-3 to work out an approximation for letter's main orientation $\boldsymbol{\theta}$;
2. Configure an interval value ***div;***
3. Configure an direction range based on θ from step 1, for example, $(\theta-15^{\circ},\theta+5^{\circ})$;
4. Starting from the minimum direction value from step 3, get the size of its directional box;
5. Iteratively change the orientation by adding ***div***, and repeat step 3 using the new direction;
6. Gather results from steps 4 and 5, if the minimum directional box occurs at either endpoint, then go back to step 3 and re-configure the interval value; otherwise report ***div*** that outputs the minimum size.

Note that ***div*** is user-defined parameter that determines precision. Smaller ***div*** offers a higher precision. Users can adjust ***div*** to achieve the best balance between ideal precision and calculation speed.

To clarify, it is not necessary to use moment theory for approximation. In some extreme cases, for example, letter *N*, when it generates a very unreliable result, orientation range in step 3 can be enlarged, which means θ is not a good reference. However, for most letters, moment theory provides a good approximation. As it helps to narrow the direction range to be tested, it is computationally effective.

Thorough examination of this feature is provided in Section 2.3. However, since the coins images are not rotationally aligned, its principal axes are not aligned either. Thus instead of comparing the principal axis itself, we choose the relative principal axis (called relative angle in this thesis) between every pair of adjacent letters, which is independent of coins directions.

2.2.4 Letter Size

Letter size consists of a pair of measurements on the letter's height and width. For upright letters, height is the distance between the highest point and the lowest point, and width is the distance between the leftmost and the rightmost points. However, this does not stand true if letters are rotated. Rather, height is measured from the top point along letter's principal axis to the bottom point, and width is measured between two extreme points along the letter's minor axis. By that definition, we can take advantage of the result from section 2.2.3. Letter size equals to the height and the width of the minimum bounding box.

Here are concrete steps on the calculation: retrieve from Section 2.2.3 the line equations of 4 edges of the minimum bounding box. Calculate 3 out of the 4 intersection points, and use Euclidean distance to get the height and the width of the box. These are the two size parameters of a letter.

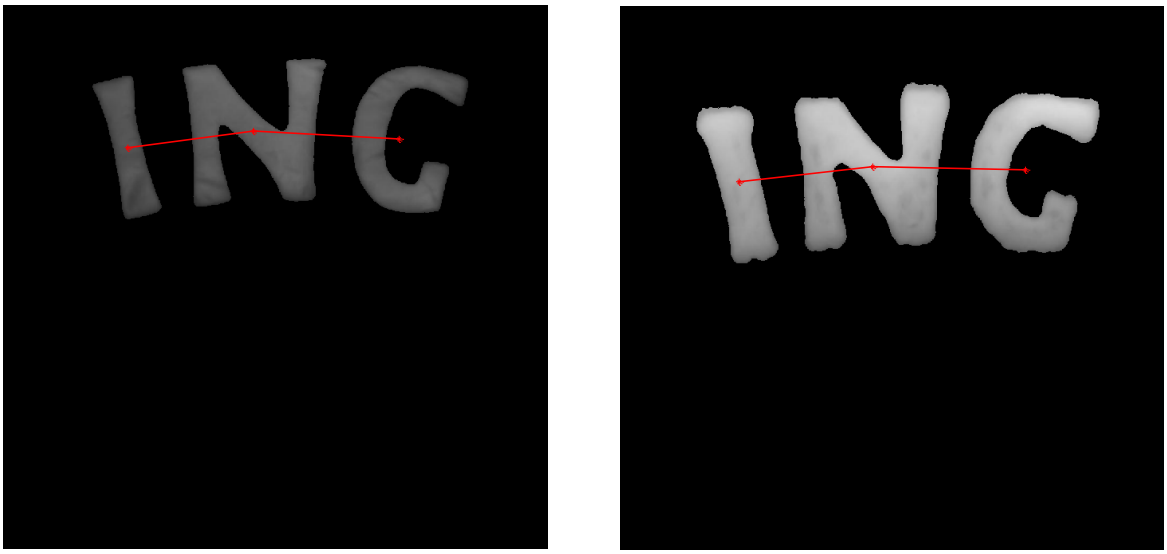
From the example shown in Fig. 2-9b, its size is 269 (pixels) for height, 226 (pixels) for width respectively. More data is provided and examined in Section 2.4.

2.2.5 Relative Distance

Unlike the previous features, relative position is an inter-lettering feature, which records the distance between every pair of adjacent letters and digits. Physically, these distances are of millimeters long, and the difference over distance is even smaller to examine. That is why even coin experts seldom use this feature as a criterion. However, with the help of *IBIS TRAX* system, we are able to obtain a much bigger and details-revealing image to help research in relative positions.

While checking distance, the two endpoints are selected to be the centers of the two letters or digits, as shown in Fig. 2-10. The method to get the centers is

described in Section 2.2.3, in moment theory, *i.e.* the first order moment divided by the total mass yields the center of a shape. As for the distance measurement, we used Euclidean distance.



a. Patch from genuine coin (Source: *90_ref_b*) **b.** Patch from fake coin (Source: *96_KT5*)

Fig. 2-10 Relative position

Fig. 2-10a is a patch extracted from a real coin surface, the distances of *I-N, N-G* are 238.96, 275.27, respectively; Fig. 2-10b is the counterpart from a fake coin. Accordingly, the distances are 254.04, 288.84 respectively.

Comprehensive comparisons can be found in section 2.4.

2.3 Classification

In this section, we shall introduce a model for differentiating genuine coins from counterfeit ones based on the features obtained from the previous sections. As a short summary for the features we get, there are six groups and each group reflects one aspect about the coin (*i.e.* stroke width, smoothness, height, width, angle and relative distance). For one coin, each feature has 29 data. In response to this distribution, we are employing a minimum spanning tree (MST) [32] based

clustering algorithm—max spacing *K*-clustering. Each feature is tested via this model, and the results are used to vote for the final decision.

Cluster analysis is an un-supervised learning method used in exploratory data analysis. It provides a means to explore and uncover the clustering structure, establish prototypes, or detect outliers. Clustering algorithms generally have natural interpretations that reveal the patterns. Some of the most well known clustering paradigms include k-means clustering, hierarchical clustering, and agglomerative clustering [29]. Unlike k-means, for agglomerative methods, clusters are detected to achieve some measure of optimality, such as the minimum intra-cluster distance or the maximum inter-cluster distance [30], and it has comparatively better performance than k-means algorithm. The MST based clustering algorithm [31, 32] is an agglomerative method, and uses the Euclidean minimum spanning tree of a graph to produce the structure of point clusters in the n-dimensional Euclidean space.

The minimum spanning tree clustering algorithm, unlike traditional clustering algorithms, does not assume a spherical shaped clustering structure [33]. It is known to be capable of detecting clusters with irregular boundaries [32]. Below are more details about the minimum spanning tree and how to reduce the MST to obtain clusters.

Minimum Spanning Tree

A spanning tree is an acyclic sub-graph of a graph G, which contains all the vertices from G. The minimum spanning tree of a weighted graph is the minimum-weight spanning tree of that graph. The minimum spanning tree is a highly abstract model designed for situations where each component gets to be wired together in

such a way that the total length or weight along this tree is the minimum. A formal definition is provided below.

Suppose a connected, undirected graph $G = (V, E)$, where V is the set of vertices, E is the set of edges, for each edge $\{(u, v) \in E | u, v \in V\}$, a weight $w(u, v)$ is assigned. The minimum spanning tree is an acyclic subset $T \subseteq E$ that connects all of the vertices and the total weight $w(T) = \sum_{(u,v)\in T} w(u, v)$ is minimized [34].

There are several known solutions to this problem, and Kruskal's algorithm is one of the approaches. Kruskal's algorithm is a greedy programming paradigm as in each consecutive step it finds a minimum spanning tree of a subset, and grows the connections till all vertices are connected.

In this project, we used a disjoint-set data structure Union-Find to implement Kruskal's algorithm. Union-Find is designed to maintain several disjoint sets of elements, by assigning the same label to only members from the class. Below is a high-level pseudo code for Kruskal's algorithm.

KRUSKAL (G, w):

1. $A = \emptyset$
2. **foreach** $v \in V$:
3. set $v.label = v$
4. sort the edges of e into non-decreasing order by weight w
5. **foreach** edge $(u, v) \in E$, taken in non-decreasing order by weight
6. **if** $u.label \neq v.label$
7. $A = A \cup \{(u, v)\}$
8. $UNION(u, v)$
9. return A

To begin with, each vertex in the graph is in a singleton set and its label is set as itself. At each minimal step, the so-far minimum weighted edge $(u, v) \in E$ is considered: if , u and v have the same label, it means they are already grouped in the same cluster and no actions are required; otherwise, merge u, v and all vertices which share either u's or v's labels, assign u's label to each member in this union. Details on implementation and the proof of correctness can be found in Kruskal's publication [35].

Max Spacing *K*-Clustering

Once the MST is established for all the given inputs, depending on the requirements, there are generally two ways to produce a set of clusters [30, 32, 36]. If the number of clusters k is given and set as a requirement, the way to get k clusters is to sort the edges of the minimum spanning tree in non-increasing order according to their weights, and remove the edges with the first *k-1* edges. Thus a complete spanning tree will get segmented into k pieces, with each piece representing one cluster. This method is also known as max spacing *K*-clustering. On the other hand, if a threshold on edge's weight is given in advance, all edges pass the threshold are removed from the tree. In this case, the number of clusters is not predictable.

In this paper, as acknowledged there are two clusters, either being counterfeit or genuine, thus we shall use the first approach where k equals to 2. The optimization goal with this approach thus is to maximize the minimum distance between any pair of points in different clusters.

This algorithm is used in section 2.4: after features feasibilities are validated, max spacing *K*-clustering is used to classify the coins.

2.4 Experiment and Data Analysis

In this section, in order to check the reliability of features extracted from section 2.2, a comprehensive examination is carried out. A valid feature should satisfy two criteria: consistency and discrepancy.

1. ***Consistency within real coins.*** For coins released in the same year, since they are using the same casting or striking die, texture, image and letterings should be the same. Thus features extracted from real coins of the same year are supposed to be close, with only small variance.
2. ***Discrepancy between real coins and fake ones***. Should a feature be useful, its value of real coin is different from that of a fake one. By convention, the difference should be at least three times of standard deviations away from the mean value of the real coins. If most letterings (more than half) meet the discrepancy criteria, it is deemed to be valid.

After the features are validated, we shall use the classifier introduced in section 2.3 to differentiate fake coins from real ones. Thus this section is divided into three parts. *First, check the consistency within real coins.* Eight real coins of year 1990 are scanned. For each coin, 29 letterings are separated, and six features are applied to each letter. *Second, check the discrepancy between real coins and fake ones.* Five coins of year 1996 are deployed, with three genuine coins and two counterfeit. And six features are extracted for 29 letterings on coin surface. Variances between genuine coins and fake coins are recorded. *Third, classify real coins and fake ones.* The same five coins used in the second part are deployed here. For each feature, the classification results are provided, and finally we shall use the voting results to give a confidence level.

2.4.1 Consistency

In this section, eight real coins are sampled (*90_Dr Suen_1, 90_De Suen_2, 90_Dr Suen_3, 90_REF_A, 90_REF_B, 90_REF_C, 90_REF_D, 90_REF_E*), and five tables are recorded to check the features' consistency; they are respectively the stroke width, smoothness, height, width, and relative distance of letterings. Comparisons are organized in a manner that for each feature, data of different digits and letters from all coins are collected in one table. For each feature, variance of each letter or digit over eight images are calculated, as shown in the last columns in Table 2-1 through Table 2-5; meanwhile an overall variance for that feature is obtained, as an indicator for the feature's effectiveness. The lower the variance is, the better the feature is.

Table 2-1 lists the tested stroke width value for each letter from different images. It can be seen that out of 29 letters, there are 9 letters having a variance larger than 5, 9 letters have a variance smaller than 1. The average variance is 3.06, and the standard deviation 1.75. The tuples in red are data responsible for larger variances. The more tuples shown in red, the less consistent the feature is. Tables 2-1 to 2-5 are organized in the same manner.

Table 2-1. Stroke Width of Lettering – Year 1990

	90_Dr Suen_1	90_Dr Suen_2	90_Dr Suen_3	90_ REF_A	90_ REF_B	90_ REF_C	90_ REF_D	90_ REF_E	MEAN	VAR
D	82.69	83.76	79.43	78.48	80.83	82.85	84.79	86.8	82.45	6.73
A	68.25	71.75	71.32	68.18	69.01	69.67	70.64	72.28	70.14	2.22
N	78.56	82.01	84.09	84.04	81.13	82.67	82.02	82.02	82.07	2.69
M	81.89	85.09	83.43	82.07	84.65	87.57	90.42	90.74	85.73	10.69
A	79.27	77.43	81.56	83.05	79.38	77.05	76.85	75.54	78.77	5.70
R	77.28	78.84	77.88	77.36	78.16	76.94	77.67	76.12	77.53	0.59
K	76.33	77.73	68.99	81.13	77.09	77.38	77.17	77.92	76.72	10.32
S	73.91	73.86	71.36	71.96	71.28	74.21	73.3	72.64	72.82	1.21
D	75.62	76.52	75.62	74.8	74.65	79.15	76.28	78.72	76.42	2.48
R	72.19	72.97	70.03	71.97	67.94	73.04	71.19	71.66	71.37	2.51
O	83.61	84.28	82.1	82.83	82.15	82	82.57	85.15	83.09	1.16
N	81.96	82.39	74.78	78.37	79.68	82.36	83.63	79.86	80.38	7.15
N	84.91	82.11	79.19	79.15	76.52	83.56	83.97	83.2	81.58	7.61
I	98.82	101.47	101.56	102.35	102.41	98.74	98.23	94.86	98.81	6.02
N	83.03	83.42	83.39	81.29	82.81	81.18	85.72	82.45	82.91	1.78
G	74.71	73.69	73.65	74.62	73.05	72.44	73.23	73.32	73.59	0.52
M	74.97	75.25	75.76	76.45	75.45	74.95	76	77.12	75.74	0.50
A	80.58	79.66	79.41	77.85	80.45	80.69	79.63	80.07	79.79	0.74
R	77.1	77.73	79.29	77.65	76.44	77.91	76.85	75.8	77.35	0.98
G	74.77	73.2	72.46	70.61	72.69	71.07	70.18	73.04	72.25	2.06
R	77.1	79.31	73.47	77.93	73.97	79	75.95	77.28	76.75	4.07
E	65.22	65.27	63.74	63.66	62.4	65.33	65.03	63.86	64.31	0.99
T	70.98	72.01	70.95	71.74	69.73	70.63	71.43	71.18	71.08	0.44
H	71.3	76.1	76.67	75.04	75.34	72.56	72.43	71.86	73.91	3.85
E	63.82	65.25	64.49	68.37	65.72	62.83	65.74	63.61	64.98	2.60
1	63.12	60.76	63.29	63.68	62.58	61.77	62.77	62.44	62.55	0.75
9	43.3	42.58	44.94	42.33	42.94	42.96	44.29	43.01	43.29	0.68
9	40.15	39.55	36.78	37.91	36.92	39.82	38.65	39.52	38.66	1.53
0	49.22	49.35	48.66	47.7	49.36	48.44	48.67	49.3	48.84	0.30

Table 2-2 shows the tested smoothness. The average variance is 0.57.

Table 2-2. Contour Smoothness – Year 1990

	90_Dr Suen_1	90_Dr Suen_2	90_Dr Suen_3	90_ REF_A	90_ REF_B	90_ REF_C	90_ REF_D	90_ REF_E	MEAN	VAR
D	30.77	31.31	27.84	30.71	30.56	30.64	30.65	30.45	30.37	0.97
A	26.06	27.91	27.20	27.77	27.85	27.21	26.94	27.32	27.28	0.32
N	28.07	30.22	30.42	30.47	29.94	28.55	29.79	29.14	29.57	0.70
M	30.33	30.83	31.17	31.54	31.08	31.36	31.75	30.90	31.12	0.17
A	32.09	29.04	30.43	28.75	30.81	30.83	30.68	30.20	30.35	0.99
R	28.38	28.72	28.20	27.22	27.95	28.44	28.24	28.29	28.18	0.17
K	28.57	29.46	25.88	28.58	28.01	29.08	29.05	28.72	28.42	1.08
S	28.18	27.61	27.23	27.72	27.00	27.69	27.63	27.33	27.55	0.11
D	32.81	32.63	33.01	32.90	31.36	31.03	33.23	32.98	32.49	0.59
R	28.62	29.87	29.12	29.56	29.00	29.59	29.80	29.56	29.39	0.16
O	31.49	30.24	30.87	31.41	30.69	31.52	31.61	31.54	31.17	0.22
N	30.57	30.50	24.98	28.54	28.83	31.22	31.47	28.15	29.28	4.03
N	29.15	27.65	24.89	28.78	28.84	29.90	31.35	30.24	28.85	3.31
I	24.51	24.83	24.56	24.51	24.46	24.65	22.52	24.59	24.33	0.48
N	27.34	28.82	28.75	28.64	28.72	28.88	27.92	28.48	28.44	0.25
G	28.45	29.12	28.66	29.25	28.29	28.68	28.51	28.15	28.64	0.13
M	31.06	32.26	31.92	32.60	32.58	30.82	32.46	32.44	32.02	0.43
A	31.01	30.42	30.41	30.41	30.57	30.19	28.69	30.35	30.26	0.40
R	27.65	28.18	28.40	27.89	28.16	28.33	27.77	28.19	28.07	0.06
G	28.64	28.59	28.27	28.44	28.17	28.57	27.68	28.56	28.37	0.09
R	29.70	30.48	28.13	30.04	29.47	29.67	29.28	29.87	29.58	0.42
E	26.58	27.51	25.87	27.37	26.50	26.60	26.09	26.64	26.64	0.28
T	29.02	26.70	28.21	28.57	28.15	27.48	29.30	29.01	28.31	0.67
H	27.52	28.03	27.49	27.24	26.79	28.03	27.62	27.73	27.56	0.15
E	26.83	26.67	25.93	25.87	25.69	25.45	25.92	25.94	26.04	0.19
1	22.00	22.09	22.24	22.38	21.91	22.44	21.99	22.23	22.16	0.03
9	18.94	18.66	18.55	18.35	19.02	19.24	18.68	18.33	18.72	0.09
9	19.12	19.20	18.64	18.93	19.04	19.02	19.11	18.89	18.99	0.03
0	20.04	19.68	19.85	20.00	19.90	19.80	19.68	19.70	19.83	0.02

Table 2-3 presents the height of letterings. The average variance is 2.68.

Table 2-3. Height of Letterings – Year 1990

	90_Dr Suen_1	90_Dr Suen_2	90_Dr Suen_3	90_ REF_A	90_ REF_B	90_ REF_C	90_ REF_D	90_ REF_E	MEAN	VAR
D	291.14	290.87	289.49	289.49	287.08	289.93	291.35	290.95	290.04	1.73
A	295.95	298.25	293.69	294.56	292.91	294.43	296.14	294.49	295.05	2.45
N	292.8	294.15	289.62	291.92	287.8	290.32	292.997	290.53	291.27	3.78
M	292.46	293.64	288.9	292.71	287.08	289.37	291.28	288.75	290.52	4.72
A	290.92	292.43	289.61	290.73	288.14	290.37	291.81	290.44	290.56	1.50
R	298.18	298.07	296.39	298.18	294.84	298.48	298.58	295.87	297.32	1.76
K	299.4	300.85	297.47	297.73	296.26	299.19	301.43	296.09	298.55	3.48
S	294	296.08	292.97	294.26	290.87	293.39	293.68	289.89	293.14	3.34
D	292.18	292.24	290.22	290.8	291.33	291.58	292.66	290.98	291.50	0.60
R	297.19	298.34	294.59	295.94	292.18	295.1	299.38	296.01	296.09	4.46
O	292.12	294.91	290.99	292.04	289.42	291	292	290.83	291.66	2.20
N	289.36	289.87	285.7	288.55	285.45	290.51	289.37	291.18	288.75	3.91
N	288.2	290.36	288.89	289.01	286.71	291.15	289.71	288.38	289.05	1.65
I	290.48	288.35	287.52	288.85	285.14	289.95		287.8	288.30	2.66
N	292.04	291.94	289.9	290.88	288.03	290.73	289.84	290.73	290.51	1.45
G	300.77	299.77	298.92	299.77	295.99	297.67	299.84	300.07	299.10	2.12
M	291.08	291.83	288.72	289.14	290.1	289.58	291.78	291.13	290.42	1.26
A	291.93	291.91	288.93	288.62	290.75	288.83	290.66	290.47	290.26	1.56
R	298.49	301.64	296.75	296.65	298.05	295.96	298.69	299.17	298.18	2.83
G	297.65	300.08	296.61	297.15	295.87	295.95	299.16	297.02	297.44	1.95
R	298.23	302.43	296.61	298.47	295.34	296.78	296.58	297.08	297.69	4.06
E	293.9	296.07	292.28	291.56	288.79	291.19	295.58	293.06	292.80	5.02
T	289.65	291.33	285.68	286.67	284.06	286.52	288.41	287.06	287.42	4.63
H	293.06	295.19	290.09	290.87	288.51	291.5	293.22	289.55	291.50	4.27
E	292.18	293.48	290.01	288.7	289.79	289.66	293.41	289.52	290.84	3.11
1	167.94	169.65	165.61	166.89	166.03	168.52	169.53	166.4	167.57	2.16
9	178.97	179.06	177	178	177.83	178.85	181.82	180.48	179.00	2.07
9	178.16	180.3	178.69	177.59	175.4	178.09	178.24	177.53	178.00	1.63
0	180.47	178.51	178.23	177.91	176.08	178.21	178.88	178.1	178.30	1.28

Table 2-4 provides the width of letters and digits. The average variance is 1.52.

Table 2-4. Width of Letterings – Year 1990

	90_Dr Suen_1	90_Dr Suen_2	90_Dr Suen_3	90_ REF_A	90_ REF_B	90_ REF_C	90_ REF_D	90_ REF_E	MEAN	VAR
D	252.23	251.91	249.14	247.82	249.17	250.61	251.55	251.90	250.54	2.36
A	251.17	251.15	251.52	250.91	250.49	251.52	249.64	249.37	250.72	0.59
N	280.50	281.84	279.34	278.29	277.49	277.67	279.73	278.13	279.12	2.02
M	331.02	334.80	330.59	330.08	329.09	332.23	331.21	328.76	330.97	3.21
A	254.87	253.02	254.18	252.23	254.16	254.68	253.30	255.55	254.00	1.03
R	225.54	226.55	224.18	224.21	223.90	227.30	225.64	227.16	225.56	1.63
K	233.88	235.47	232.99	233.46	231.95	234.98	233.68	235.56	234.00	1.40
S	211.61	212.64	208.64	209.70	208.80	211.56	211.84	209.57	210.55	2.07
D	247.22	250.79	247.59	246.69	247.54	246.71	247.48	247.10	247.64	1.52
R	228.44	228.87	225.33	226.34	228.26	226.95	228.54	231.52	228.03	3.08
O	253.52	256.46	252.55	255.97	252.77	253.00	255.00	253.00	254.03	2.10
N	287.92	289.61	289.56	290.23	286.93	286.17	288.17		288.37	1.94
N	281.56	283.97	282.62	282.62	282.36	284.46	283.03	280.53	282.64	1.37
I	96.09	97.31	97.00	96.26	96.26	98.22		96.83	96.85	0.48
N	280.32	282.56	281.90	280.12	280.91	282.05	280.21	280.70	281.10	0.78
G	211.41	214.91	211.56	212.48	210.68	213.34	212.35	210.11	212.11	2.05
M	338.74	338.65	338.13	336.53	336.76	336.66	337.17	336.37	337.38	0.84
A	251.38	250.83	249.16	249.21	251.60	250.79	251.59	250.67	250.65	0.83
R	221.67	223.28	220.71	220.09	221.10	220.35	220.07	220.78	221.01	0.99
G	211.20	212.33	216.20	211.64	211.23	212.23	212.84	211.70	212.42	2.32
R	231.91	232.82	229.54	231.73	229.35	232.73	233.64	230.35	231.51	2.23
E	197.06	197.35	193.38	195.50	193.05	195.54	195.76	196.46	195.51	2.17
T	237.32	237.30	235.49	237.42	236.16	235.86	236.63	235.40	236.45	0.61
H	254.57	255.26	252.05	253.34	251.75	253.80	252.62	253.01	253.30	1.28
E	198.66	199.09	193.53	195.56	196.25	197.62	196.28	198.51	196.94	3.12
1	72.98	72.88	72.51	71.59	71.87	72.44	73.34	74.11	72.72	0.57
9	127.88	128.20	127.00	127.00	127.79	127.76	128.37	128.82	127.85	0.35
9	118.57	119.79	119.55	118.17	119.19	118.69	120.36	120.35	119.33	0.59
0	150.10	149.58	148.80	149.09	150.33	149.61	149.58	151.27	149.80	0.52

Table 2-5 lists the relative distance between pairs of letters or digits. The average variance is 0.53.

Table 2-5. Relative Distance – Year 1990

	90_Dr Suen_1	90_Dr Suen_2	90_Dr Suen_3	90_ REF_A	90_ REF_B	90_ REF_C	90_ REF_D	90_ REF_E	MEAN	VAR
D-A	274.32	274.35	274.31	273.77	272.83	272.83	274.04	273.73	273.77	0.35
A-N	289.70	290.75	290.38	289.07	289.21	289.49	290.22	288.91	289.72	0.39
N-M	345.31	346.93	346.64	347.58	346.50	347.61	347.70	347.58	346.98	0.60
M-A	333.70	333.73	332.34	332.00	333.32	333.77	332.95	333.08	333.11	0.39
A-R	269.10	270.42	268.95	270.41	268.91	270.66	270.11	270.91	269.93	0.59
R-K	269.73	269.91	268.96	269.22	269.12	270.51	270.11	269.91	269.68	0.25
K-S	269.39	268.54	268.69	270.10	268.84	269.76	268.60	269.69	269.20	0.32
S-D	345.99	347.87	347.13	345.08	345.26	345.94	345.23	345.79	346.04	0.85
D-R	269.32	270.55	269.66	267.57	267.97	269.15	269.40	269.54	269.14	0.79
R-O	271.09	270.81	269.98	270.85	270.53	270.94	270.17	271.49	270.73	0.21
O-N	293.15	295.67	295.26	296.14	294.74	295.85	294.19	298.45	295.43	2.13
N-N	335.35	333.25	333.65	334.42	333.71	334.09	332.88	331.12	333.56	1.35
N-I	233.93	233.59	233.00	233.98	233.17	233.78	235.71	233.35	233.81	0.62
I-N	234.95	235.36	236.10	235.64	235.51	234.45	234.27	235.21	235.19	0.33
N-G	275.77	277.40	275.19	275.13	275.42	276.60	275.52	274.44	275.68	0.74
M-A	329.53	330.23	329.84	330.37	329.67	329.51	331.01	329.90	330.01	0.23
A-R	271.23	269.85	268.60	270.30	269.92	270.74	269.96	271.32	270.24	0.68
R-G	252.19	252.91	253.29	251.79	251.25	252.16	252.80	253.28	252.46	0.47
G-R	260.74	261.09	260.54	261.63	261.33	261.01	261.58	261.60	261.19	0.15
R-E	249.91	250.35	250.47	249.27	248.45	250.11	250.70	248.65	249.74	0.63
E-T	237.91	236.89	237.43	238.21	237.82	237.66	237.24	237.38	237.57	0.15
T-H	259.75	257.42	259.38	258.40	258.13	258.41	256.83	257.45	258.22	0.87
H-E	262.58	263.87	262.36	263.03	263.03	262.75	264.05	262.71	263.05	0.32
1-9	126.73	126.66	127.09	126.64	126.10	126.98	127.41	126.99	126.83	0.13
9-9	143.69	143.61	143.49	143.54	143.69	144.14	143.42	144.71	143.79	0.16
9-0	148.99	149.26	148.92	148.64	148.67	149.16	149.65	148.61	148.99	0.11

2.4.2 Discrepancy

As stated, discrepancy property means that the features extracted from real coins and fake coins are different. It is a significant aspect for a valid feature.

For discrepancy check, we sampled 5 coins, among which three are real coins (*96_ref_a, 96_ref_real, 96_Dr Suen_1*), and two are fake ones (*96_KT3, 96_KT5*). In this section, six tables (Table 2-6 to Table 2-11) are recorded to check feature discrepancy; they are respectively stroke width, smoothness, height, width, relative distance, and relative angle of Letterings. SD $\boldsymbol{\sigma}$ stands for standard deviation. Comparisons are organized in a manner that for each feature, data of different letters and digits and image sources is collected. For each feature, the mean value and standard deviation of each letters and digits over three real images are calculated. Standard deviation can be seen as an indicator of the feature's effectiveness. Meanwhile features extracted from fake coins are compared with mean and variance. Features extracted from fake coins that are $5\boldsymbol{\sigma}$ away from the mean value are marked in blue, whereas those $10\boldsymbol{\sigma}$ away are shown in red. Obviously, the more data marked in blue or red, the more effective the feature is.

Table 2-6 shows stroke width of letterings.

Table 2-6. Stroke Width of Lettering – Year 1996

	96_ ref_a	96_ ref_real	96_Dr Suen_1	Mean	SD σ	96_KT3	96_KT5
D	62.85263	61.14388	65.98182	63.32611	2.003	74.77842	68.27536
A	58.1962	54.81967	56.14286	56.38624	1.389	63.70925	
N	70.32877	68.56643	65.46875	68.12132	2.009	69.68643	69.44511
M	66.16162	64.51429	68.41111	66.36234	1.597	78.95	75.70569
A	52.62637	55.34444	56.83562	54.93548	1.743	60.86525	66.72911
R	61.67491	65.34297	61.66373	62.89387	1.732	67.0303	68.66751
K	67.73118	71.0443	66.69582	68.49043	1.855	69.61348	71.34091
S	57.47143	58.29252	58.47727	58.08041	0.437	70.30097	68.99597
D	60.29801	59.59091	60.77907	60.22266	0.488	76.85432	72.78136
R	58.28763	56.97321	59.48829	58.24971	1.027	67.49853	65.38679
O	60.71196	60.73333	60.62657	60.69062	0.046	53.82773	56.51831
N	68.91705	61.45455	67.84892	66.07351	3.295	84.74783	79.91781
N	73.38197	75.5202	74.23237	74.37818	0.879	85.25781	88.32609
I	79.58333	87.2766	77.8	81.55331	4.112	94.14103	100.5909
N	72.68132	70.07143	71.62871	71.46049	1.072	84.16456	86.19615
G	59.9359	63.93243	59.9881	61.28548	1.872	72.63694	76.40625
M	56.92308	60.20556	58.68644	58.60502	1.341	63.97037	64.975
A	61.89189	62.84211	61.7967	62.1769	0.472	64.47872	67.02439
R	70.08108	73.81308	72.94118	72.27845	1.594	72.28571	66.15837
G		57.24242	57.78431	57.51337	0.271	57	59.67788
R	65.14516	64.21809	65.37437	64.91254	0.500	68.0987	67.61153
E	53.66531	54.02439	54.20718	53.96563	0.225	52.76684	54.35057
T	54.06812	55.10137	54.49441	54.55463	0.424	53.40397	52.61345
H	59.34091	62.06048	61.07947	60.82695	1.125	60.33077	60.51534
E	51.4245	53.75135	51.54942	52.24176	1.069		
1		42.10909	45.93939	44.02424	1.915	46.4129	51.63934
9		40.61538	40.12268	40.36903	0.246	40.14234	40.06731
9		40.36015	40.53293	40.44654	0.086	42.8125	45.5969
6	39.64571	39.33871	39.37367	39.4527	0.137	44.65686	49.03727

Table 2-7 shows contour smoothness as defined in section 2.2.2.

Table 2-7. Contour Smoothness – Year 1996

	96_ ref_a	96_ ref_real	96_Dr Suen_1	Mean	SD σ	96_KT3	96_KT5
D	22.00192	22.11326	21.90108	22.00542	0.087	22.74063	21.60513
A	21.43191	21.70021	20.88145	21.33786	0.341	23.34019	
N	23.68136	23.9508	23.14828	23.59348	0.333	20.62135	21.1816
M	25.33111	25.33651	25.03154	25.23305	0.143	26.04237	27.05901
A	22.31111	22.70117	22.35462	22.45563	0.175	24.66741	25.37519
R	20.9176	21.46281	20.61757	20.99933	0.350	25.19836	25.01384
K	21.28271	21.75574	21.53751	21.52532	0.193	24.69078	24.90033
S	20.79252	21.19795	21.06676	21.01908	0.169	26.0885	25.31497
D	24.55256	24.63761	24.85146	24.68054	0.126	29.94774	29.27457
R	22.55208	22.50773	22.68653	22.58211	0.076	24.80579	24.65302
O	23.16667	23.30159	23.34798	23.27208	0.077	22.7041	23.38383
N	25.71376	23.01438	25.81779	24.84864	1.298	29.37978	28.87407
N	24.84696	25.04263	24.69293	24.86084	0.143	31.28564	30.87415
I	20.31023	20.34949	20.29257	20.31743	0.024	24.85699	25.31461
N	24.76143	24.04155	24.36842	24.39047	0.294	25.35752	28.35794
G	23.55894	24.19081	23.18851	23.64608	0.414	26.23857	26.70344
M	25.78876	26.09744	25.81575	25.90065	0.140	26.6718	27.98324
A	23.9084	24.36725	23.74567	24.00711	0.263	22.97519	24.47543
R	23.01158	23.29974	23.22344	23.17825	0.122	21.1161	21.87085
G		22.56122	22.9724	22.76681	0.206	19.84211	19.5282
R	22.65786	22.94013	22.90979	22.83593	0.127	23.20717	24.1451
E	19.3616	20.91045	20.40799	20.22668	0.645	19.59217	19.51005
T	22.78294	22.67784	22.21996	22.56025	0.244	21.82833	21.59537
H	22.24078	21.8365	22.44339	22.17355	0.252	21.67481	19.86398
E	19.19444	19.29046	19.52946	19.33812	0.141		
1		17.8316	17.55809	17.69485	0.137	17.90258	18.71785
9		16.70851	16.35095	16.52973	0.179	17.51757	18.25503
9		16.31563	16.46878	16.3922	0.077	18.57906	19.27959
6	15.64627	16.04812	15.59459	15.76299	0.203	18.72268	19.39615

Table 2-8 and Table 2-9 demonstrate the height and width of lettering.

Table 2-8. Height of Lettering – Year 1996

	96_ ref_a	96_ ref_real	96_Dr Suen_1	Mean	SD σ	96_KT3	96_KT5
D	266.19	265.84	265.73	265.92	0.196	266.1	267.79
A	272.17	273.32	272.45	272.6467	0.490	276.51	
N	266.81	267.27	267.17	267.0833	0.1985	272.71	269.64
M	269.7	269.32	268.56	269.1933	0.474	275.71	276.21
A	264.81	264.17	264.65	264.5433	0.272	277.68	275.33
R	273.85	273.47	272.73	273.35	0.465	288.31	288.51
K	279.7	280.37	278.15	279.4067	0.930	301.55	302.78
S	277.13	276.87	275.77	276.59	0.589	300.24	299.36
D	268.36	270.18	269.49	269.3433	0.750	297.46	295.78
R	280.77	278.92	279.8	279.83	0.756	304.58	307.73
O	274.73	275.34	274	274.69	0.548	298	294
N	265.83	268.96	265.8	266.8633	1.483	280.26	285.36
N	271.01	272.44	274.02	272.49	1.229	295.9	302.18
I	270.36	271.88	271.02	271.0867	0.622	299.37	304.52
N	271.06	272.43	272.59	272.0267	0.687	304.33	311.27
G	282.36	284.99	283.62	283.6567	1.074	305.89	312.69
M	271.23	271.2	270.71	271.0467	0.238	281.75	283.27
A	271.6	272.32	271.3	271.74	0.428	284.01	280.28
R	279.54	280.66	282.77	280.99	1.339	285.08	280.39
G		280.23	282.17	281.2	0.97	280.09	276.39
R	280.72	280.05	279.02	279.93	0.699	287.69	286.73
E	273.02	274.49	273.13	273.5467	0.669	277.27	279.295
T	269	269.09	267.06	268.3833	0.936	259.73	260.12
H	269.1	269.54	269.29	269.31	0.180	260.44	
E	269.61	270.19	269.98	269.9267	0.240		
1		187.51	189.48	188.495	0.985	196.35	198.47
9		199	198	198.5	0.5	203.2	206
9		199.24	198.28	198.76	0.48	206.16	208
6	198.14	198.28	197.34	197.92	0.414	214.57	218.38

Table 2-9. Width of Lettering – Year 1996

	96_ ref_a	96_ ref_real	96_Dr Suen_1	Mean	SD σ	96_KT3	96_KT5
D	226.2	223.02	224.74	224.65	1.300	231.52	229.94
A	228.74	227.99	227.85	228.19	0.391	235.76	
N	265.43	266.42	264.58	265.48	0.752	270.91	271.01
M	317.28	315.01	313.62	315.30	1.509	315.33	316.1
A	229.98	227.55	225.78	227.77	1.722	233.46	230.58
R	210.92	206.73	207.7	208.45	1.791	214.23	213.17
K	209.05	211.79	211.75	210.86	1.282	221.86	219.68
S	186.89	188.57	189.88	188.45	1.224	194.4	199.7
D	228.94	230.43	224.66	228.01	2.446	252.6	252.93
R	210.59	213.06	211.43	211.69	1.025	224.53	224.52
O	233.05	232.47	231	232.17	0.863	228	231
N	273.12	270.12	272.21	271.82	1.256	294.45	295.996
N	274.38	273.69	271.57	273.21	1.196	302.89	304.58
I	85.5	85.02	83.23	84.58	0.977	105.73	107.53
N	276.15	276.28	275.26	275.90	0.453	288.15	290.35
G	200.79	200.47	199.45	200.24	0.571	216.86	216.95
M	317.29	320.31	318.7	318.77	1.234	311.22	310.82
A	229.63	226.46	228.83	228.31	1.346	230.51	231.81
R	209.46	210.22	208.73	209.47	0.608	207.89	210.39
G		204.43	200.27	202.35	2.080	199.64	210.58
R	214.28	212.52	211.05	212.62	1.320	216.83	214.98
E	187.66	182.63	180.87	183.72	2.877	181.04	176.897
T	224	226.72	226.05	225.59	1.157	226.92	226.69
H	242.56	243.22	243.25	243.01	0.318	242.82	
E	179.6	179.41	179.53	179.51	0.078		
1		67.59	67.13	67.36	0.230	70.83	74.98
9		121	123	122.00	1.000	121.63	122
9		123.51	122.93	123.22	0.290	126.75	128
6	121.58	121.45	120.98	121.34	0.258	128.97	133.25

Table 2-10 provides the relative distance between adjacent letters and digits.

Table 2-10. Relative Distance – Year 1996

	96_ref_a	96_ ref_real	96_Dr Suen_1	Mean	SD σ	96_KT3	96_KT5
D-A	274.8961	275.2098	275.4559	275.1872	0.229	282.2989	
A-N	294.4393	295.2439	295.1766	294.9533	0.364	295.215	
N-M	353.0188	353.4682	353.7443	353.4104	0.299	353.6425	355.828
M-A	335.5109	334.2671	334.3847	334.7209	0.561	330.7091	330.5215
A-R	273.1517	272.1756	272.5851	272.6374	0.400	273.7566	272.8
R-K	272.9589	274.3168	274.4947	273.9235	0.686	271.9181	271.0041
K-S	272.9129	274.4736	274.9159	274.1008	0.859	272.9242	273.5124
S-D	350.2112	349.9814	347.8079	349.3335	1.083	356.8053	357.5165
D-R	270.6678	271.9751	269.3765	270.6731	1.061	274.2744	274.5802
R-O	273.9351	274.5075	273.815	274.0859	0.302	284.7991	284.969
O-N	297.1044	295.3523	297.1875	296.5481	0.846	310.0245	310.8316
N-N	336.3342	335.866	336.9023	336.3675	0.424	361.9551	361.2374
N-I	232.584	233.3513	232.853	232.9294	0.318	243.0236	242.3296
I-N	238.2065	238.6463	238.6335	238.4954	0.204	252.3719	253.3257
N-G	276.9527	276.5767	277.3185	276.9493	0.303	288.109	288.1365
M-A	329.3137	331.6395	331.2331	330.7288	1.014	325.5001	325.6522
A-R	272.2706	272.8065	273.087	272.7214	0.339	268.9183	270.7229
R-G		257.1475	255.063	256.1053	1.042	252.3316	257.3268
G-R		267.062	268.2914	267.6767	0.615	262.9235	258.048
R-E	253.3859	252.3999	251.297	252.3609	0.853	247.742	248.7626
E-T	243.3407	245.1402	246.3138	244.9316	1.223	246.1455	243.25
T-H	261.079	261.6539	262.2279	261.6536	0.469	263.5279	262.5217
H-E	263.2211	263.9526	263.623	263.5989	0.299		
1--9		147.4919	147.6096	147.5508	0.059	150.8265	151.069
9--9		180.4147	180.7163	180.5655	0.151	179.267	179.2042
9--6		177.9654	177.575	177.7702	0.195	185.7908	186.2343

Table 2-11 gives relative angles between adjacent letters and digits on coin surface.

Table 2-11. Relative Angle – Year 1996

	96_ref_a	96_ ref_real	96_Dr Suen_1	Mean	SD σ	96_KT3	96_KT5
D-A	13.34	13.27	13.33	13.31333	0.031	8.83	
A-N	12.41	12.98	12.34	12.57667	0.287	13.78	
N-M	15.39	15.19	15.49	15.35667	0.125	15.3	15.708
M-A	13.94	13.67	13.25	13.62	0.284	15.03	14.962
A-R	10.95	10.32	11.07	10.78	0.329	11.6	11.56
R-K	9.13	9.52	10.03	9.56	0.369	8.43	8.6
K-S	13.28	12.02	11.65	12.31667	0.698	13.87	13.33
S-D	14.42	15.86	16.12	15.46667	0.748	16.51	16.96
D-R	10.03	10.44	10.05	10.17333	0.189	6.49	6.84
R-O	13.09	12.73	12.558	12.79267	0.222	17.17	12.25
O-N	13.55	13.64	13.27	13.48667	0.158	13.81	19.21
N-N	14.3	14.04	14.4	14.24667	0.152	13.91	11.76
N-I	9.96	10.52	9.93	10.13667	0.271	10.43	12.2
I-N	9.84	9.31	10.11	9.753333	0.332	7.92	7.92
N-G	15.83	15.97	15.38	15.72667	0.252	16.2	16.38
M-A	13.85	13.61	13.92	13.79333	0.133	15.22	14.878
A-R	8.55	8.96	8.75	8.753333	0.167	8.2	9.322
R-G		18.03	14.77	16.4	1.630	13.97	15.51
G-R		4.83	8.52	6.675	1.845	7.89	5.41
R-E	10.23	10.55	10.46	10.41333	0.135	10.01	9.46
E-T	11.3	12.29	11.27	11.62	0.474	15.08	15.42
T-H	10.3	9.66	10.66	10.20667	0.414	6.9	
H-E	11.97	10.93	11.46	11.45333	0.425	83.76	
1--9		-8.61	-4.64	-6.625	1.985	-3.65	-12.558
9--9		-3.52	-8.71	-6.115	2.595	-9.49	0.108
9--6		-8.66	-7.4	-8.03	0.630	-5.69	-7.38

2.4.3 Classification Results

In this section, we have built a max spacing 2-clustering classifier according to section 2.3. Table 2-12 records the clustering results: for the first five features, *96_KT3* and *96_KT5* are classified in the same group whereas the remaining coins in another group; for the feature *Relative Angle,* the results are slightly different, that *96_KT5* is grouped in one cluster and the rest in the other.

Table 2-12. Classification Based on Max Spacing *K*-clustering

Feature	Clustering results		Max spacing	
	Cluster A	Cluster B	Min Inter Spacing σ_1	Max Intra Spacing σ_2
Stroke width	96_ref_a, 96_ref_real, 96_Dr Suen_1	96_KT3, 96_KT5	8.23	3.47
Smoothness	96_ref_a, 96_ref_real, 96_Dr Suen_1	96_KT3, 96_KT5	2.85	0.96
Char. Height	96_ref_a, 96_ref_real, 96_Dr Suen_1	96_KT3, 96_KT5	16.03	3.52
Char. Width	96_ref_a, 96_ref_real, 96_Dr Suen_1	96_KT3, 96_KT5	11.23	3.12
Rel. Dist.	96_ref_a, 96_ref_real, 96_Dr Suen_1	96_KT3, 96_KT5	8.20	1.74
Rel. Ang.	96_ref_a, 96_ref_real, 96_Dr Suen_1, 96_KT3,	96_KT5	2.25	2.10

Note min inter spacing σ_1 is the minimum distance for any pair of samples from different classes, and also is the output of max spacing *K*-clustering algorithm; this index indicates how far two clusters distribute in feature space. Max intra spacing σ_2 is the maximum distance between any pair of samples from the same cluster; it is an index of how close the samples are in the same cluster. A feature is most

effective when the σ_1 is far larger than max intra-spacing σ_2. Here we adopt normalized $\Delta = \sigma_1/\sigma_2$ as voting weight. For future test, its classification confidence can be calculated based on Eq. 2-4: i is the index of feature, $\bar{\Delta}_i$ is normalized Δ_i.

$$c = \sum_{i=1}^{6} \bar{\Delta}_i \cdot r_i, \quad r_i = \begin{cases} 1, cluster\ A \\ 0, cluster\ B \end{cases} \qquad 2\text{-}4$$

By rule of voting, *96_KT5* and *96_KT3* are separable from the other coins. The result corresponds to our prior knowledge and validates the feasibility of the method. All the new testing data clustered in the same category as *96_KT5* or *96_KT3* shall be marked as fake, otherwise shall be marked as genuine.

2.5 Conclusion

This chapter focuses on features from individual letters. By checking two aspects of a feature, consistency and discrepancy, we find the features are valid and can contribute to the detection. By deploying the max spacing *K*-clustering classifier, we have proved that the genuine coins and fake coins are separable based on the features extracted. Thus this classifier contributes as a solid approach to detect fake coins. Out of the six features, the height and width of lettering and relative distance are in general more effective since they possess a very low variance among genuine coins and high discrepancy between genuine and fake coins.

Chapter 3 Image Analysis

Image analysis is a very broad topic. In this chapter, we are focusing on one aspect – the shape of the head profile on coin surface. There has been quite a lot of research on this topic. To name a few, the medial axis transform (MAT), primitive shape decomposition, and moment-based approaches [37]. In those methods, the shape is usually preserved in a chain of descriptors that can be used for the purpose of shape reconstruction, recognition, or representation. However, those algorithms are inappropriate for our project. For example, chain code, though perfect for information preservation, its expression is not straightforward; moment-based methods are not sensitive enough for very similar shapes.

Thus in this thesis, we propose a novel feature that can preserve and represent shapes. Below are the detailed steps used in this chapter.

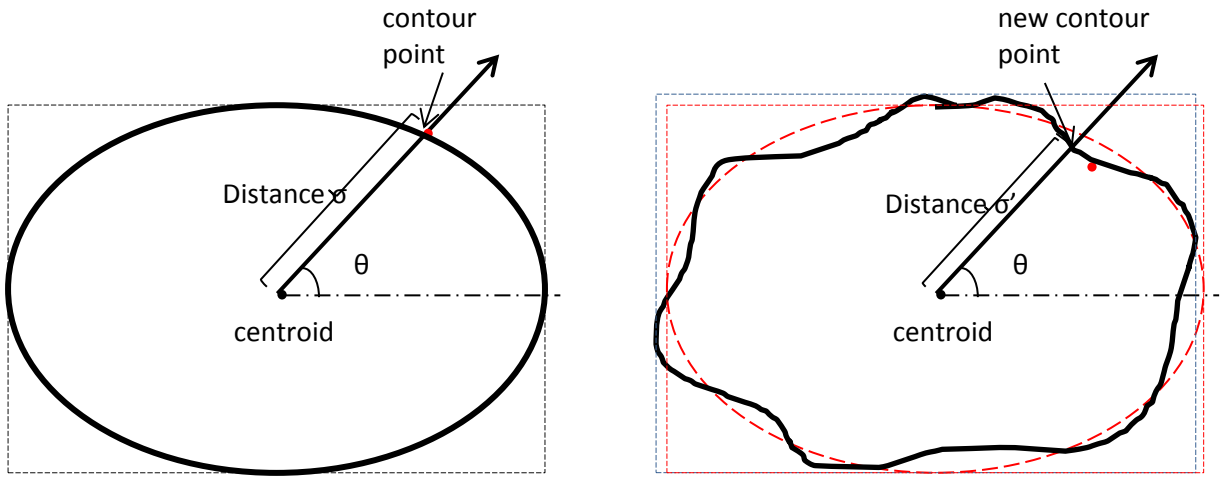
1. ***Pre-processing:*** to extract the image centroid and edges;
2. ***Coordinate converting***: to convert the edge image in Cartesian coordinates to polar coordinates;
3. ***Contour extraction:*** to extract contour from either binary image, or edge image under polar coordinates.
4. ***Dissimilarity measurement***: to quantize their difference and compare shapes.

In the following sections, the feature is defined and its validation elaborated in section 3.1. In section 3.2, we shall discuss two methods to obtain the contour of head profile. Section 3.3 shows the experimental results.

3.1 A Novel Shape Feature '*angle-distance*'

The feature is named '*angle-distance*'. It is a vector of size N by 1, where N stands for the number of subdivisions equally split in an intact circle plate.

Normally N equals to 360, meaning the circle plate is split into 360 bins with each bin representing 1 degree. In Fig. 3-1, shooting a ray of angle θ from the centroid of the shape, the distance between the intersection on the contour and the centroid is σ. For a ray shooting from each angle θ, there is one σ associated. That is the novel '*angle-distance*' feature, *i.e.* σ values in N directions.



a. shape 1 **b.** shape 2

Fig. 3-1 θ-σ correspondence

3.1.1 Validation

If a shape is a closed curve, or almost a closed curve, it is possible to be transformed from a two-dimensional representation to a one-dimensional vector. As shown in Fig. 3-1a, for an arbitrary shape, assume its centroid and boundary are explicit. Radiate from the centroid, then for any particular direction θ**,** there would be one contour point corresponding to it. If the shape is not rotated, the distance between the contour point and the centroid is fixed, *i.e.* there is a constant correspondence between angle θ and distance σ. Thus this correspondence can be employed as a feature to represent and reconstruct the shape.

The main benefits of deploying θ-σ correspondence are three.

1. *Unique descriptor.* As shown in Fig. 3-1b, assume that two shapes have already been aligned. If the testing shape, as shown in black, is slightly different from the template shape shown in red dash, the distance value σ at the same direction θ will be different.
2. *Better and easier quantization.* The more similar two shapes are, the less different their vector descriptors are. Thus we can easily use Euclidean distance to measure how different two shapes are.
3. *Rotational invariance.* Rotation is a main concern for shape registration. Under a Cartesian coordinate system, if a shape is rotated, it would be hard to track the previous-and-afterwards pixel correspondence. However, in the polar system, this is a lot easier to deal with. A mathematical proof is provided in next section.

3.1.2 Conversion of Coordinates

Since the θ-σ correspondence resembles the definition of polar coordinates, we obtain the feature by converting shapes from Cartesian coordinates to polar coordinates. Polar coordinates record points by a pair of measurements: distance σ from the pixel to the pole, and angle θ from the polar axis.

Converting coordinates (x, y) in Cartesian system to (σ, θ) in polar system can be implemented through Eq. 3-1 and Eq. 3-2.

$$\sigma = \sqrt{x^2 + y^2} \quad r \geq 0 \tag{3-1}$$

$$\theta = \begin{cases} \arctan\left(\frac{y}{x}\right), & x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi, & y \geq 0, x < 0 \\ \arctan\left(\frac{y}{x}\right) - \pi, & y < 0, x < 0 \\ \frac{\pi}{2}, & y > 0, x = 0 \\ -\frac{\pi}{2}, & y < 0, x = 0 \\ undefined, & y = 0, x = 0 \end{cases} \quad \text{3-2}$$

Based on this transformation, for an image *f(x,y)*, its converted image in polar coordinate is *g(σ,θ)*. Pixels that correspond according to Eq. 3-1 and Eq. 3-2 share the same intensity. Fig. 3-2 shows the converted image in polar coordinate.

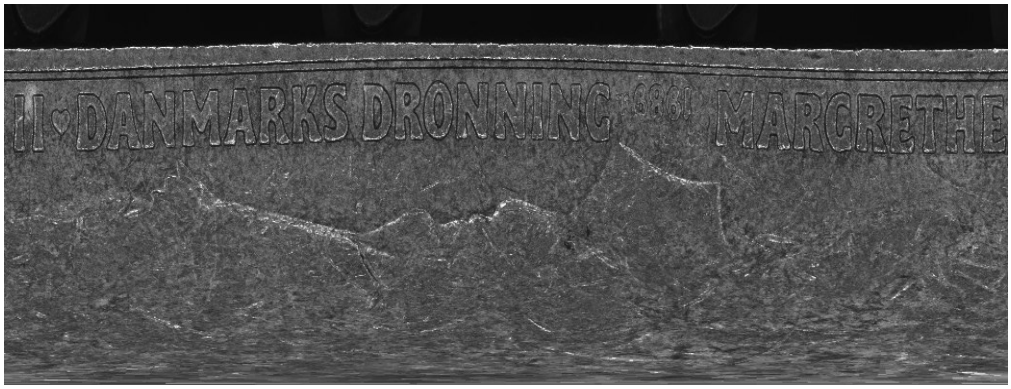


Fig. 3-2 Edge image in polar system

To note that in this project, all the edge images are converted by using horizontal axis as the polar axis. Though the orientations of images are not optimally aligned. However, this will not be a problem as explained below.

For an arbitrary point *(x, y)*, its polar coordinates *(σ, θ)*, the relation between two coordinates is shown in Eq. 3-1 and Eq. 3-2.

Suppose that a shape is rotated by angle *α*, then the new coordinates *(x', y')* and *(σ', θ')* will transfer as shown in Eq. 3-3 to 3-5.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\alpha & -sin\alpha \\ sin\alpha & cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} cos\alpha \cdot x - sin\alpha \cdot y \\ sin\alpha \cdot x + cos\alpha \cdot y \end{bmatrix} \quad 3\text{-}3$$

$$\begin{aligned} \sigma' &= \sqrt{(\cos\alpha \cdot x - \sin\alpha \cdot y)^2 + (\sin\alpha \cdot x + \cos\alpha \cdot y)^2} \\ &= \sqrt{(\cos^2\alpha + \sin^2\alpha) \cdot x^2 + (\cos^2\alpha + \sin^2\alpha) \cdot y^2} \\ &= \sqrt{x^2 + y^2} = \sigma \end{aligned} \quad 3\text{-}4$$

$$\begin{aligned} \tan\theta' &= \frac{\sin\alpha \cdot x + \cos\alpha \cdot y}{\cos\alpha \cdot x - \sin\alpha \cdot y} \\ &= \frac{\sin\alpha \cdot x + \cos\alpha \cdot (x \cdot \tan\theta)}{\cos\alpha \cdot x - \sin\alpha \cdot (x \cdot \tan\theta)} \\ &= \frac{\tan\alpha + \tan\theta}{1 - \tan\alpha \cdot \tan\theta} = \tan(\theta + \alpha) \end{aligned} \quad 3\text{-}5$$

Thus, if a shape is rotated, its distance value σ will not change. And θ' equals its old angle θ plus the rotation angle α. This property implies that for the *angle-distance* feature, by shifting the values in this vector, we are rotating the shape cyclically.

To be more concrete, suppose the vector obtained is

$$a^T = [a_1, a_2, \dots, a_{360}]$$

If the shape is rotated clockwise by α degree, what need to be done with the feature vector is to shift all values by α place, *i.e.*

$$a^T \rightarrow a(\alpha)^T = [a_{361-\alpha}, \dots, a_{360}, a_1, \dots a_{360-\alpha}]$$

3.1.3 Dissimilarity Measurement

In this project, we adopted Euclidean distance as the difference between two shapes. The rule of thumb is that the larger the distance is, the more different the two shapes are. Meanwhile, as notified in section 3.1.2, misalignment should also

be considered. Here is the strategy, suppose the feature vectors of two shapes are *a* and *a'* respectively, we define a measurement *c(α)*.

$$c(\alpha) = \sqrt{\frac{(a(\alpha) - a')^T (a(\alpha) - a')}{length(a)}} \quad 3\text{-}6$$

Physically, *c(α)* means on average what is the difference between two shapes with the first shape rotated α degrees. Given the rotational invariance property of the feature, and the fact that two shapes to be compared can be misaligned, we adjust Eq. 3-6 to a dissimilarity measurement *k(a, a')*.

$$k(a, a') = \min_\alpha c(\alpha) \quad 3\text{-}7$$

k(a, a') will be used to test how dissimilar two shapes are. Meanwhile, we tested *g(a,a')* for misaligned degrees between two shapes.

$$g(a(\alpha), a') = \alpha_{min} = arg\min_\alpha c(\alpha) \quad 3\text{-}8$$

3.2 Contour Extraction

Section 3.1 described how to get the *angle-distance* feature from a known shape. As another important component of this application pipeline, this section discusses how to extract contours from coin images. Generally speaking, there are two ways to process it: the first is to separate images as foreground from coin surfaces, and the second is to detect edges of images and track down edges in polar coordinates. The first part of this section illustrates how separation is done, while the second part elaborates how to track down edges in polar coordinates.

3.2.1 Separating Foreground

In this section, the image of head profile is referred as the foreground, whereas the plain surface of coin is regarded as the background.

Fig. 3-3 shows the workflow of separating the foreground and surface.

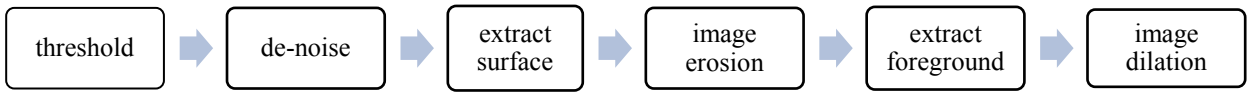
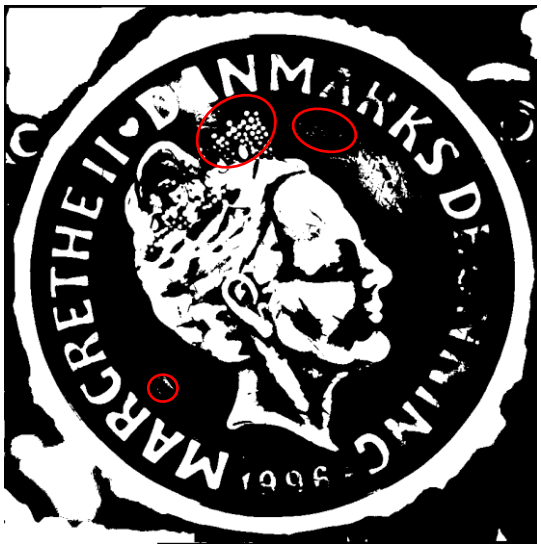


Fig. 3-3 Workflow of image extraction

Given the original depth image Fig. 3-4a, it is obvious that the areas close to image stamp and letterings are much brighter than the coin surface, thus it is proper to apply Otsu, a global binary method. Fig. 3-4b shows the rough result, where the decorations on the crown are scattered, and noises spread across the coin surface. In response to the unwanted details and noises, cleaning of small pieces is implemented, and the result is shown in Fig. 3-4c. However, it is still notable that design of space made the relatively dark areas on face and hair hollow out, as marked in red ovals. Besides the circumference is contributing noise. To get rid of most noises, the coin surface is extracted, as shown in Fig. 3-4d the white area. So far the head profile is still not complete. Alternatively, it can be regarded as the coin surface is contaminated by part of the head stamp, see the read ovals. In response, two morphology techniques are applied, erosion and dilation.

a. Original Image



b. Binary image with noise

c. Circumference noise **d.** Head profile eroded by surface

Fig. 3-4 Head profile binarization

Erosion and Dilation

Morphology is a powerful tool for analyzing and processing geometrical structures and shapes. Developed upon set theory, objects in images are represented as sets. Mathematically speaking, a set is a combination of distinct objects. For example, the set of all white pixels is a complete description of a binary image. Each element of the set is a tuple of coordinates. Another key component of operation is a pre-defined probe, called *structuring elements* (SEs). It is defined as a sub-image. Fig. 3-5a shows two typical SEs. Each grid means one pixel, whereas the dot denotes the origin.

a. SE **b.** Illustration

Fig. 3-5 Erosion example

The basic idea is to probe an image under certain conditions. Fig. 3-5b demonstrates how SE functions. Suppose an operation is defined on set *A*, using structure SE as shown (center at lower right corner): generate a new set *B* by probing SE over set *A* so that the origin of SE traverses all elements in set *A*; at each location of *A*, if and only if SE is completed contained in set *A*, mark that location for the new set *B*, as shown in Fig. 3-5b.

The procedure used in this example is called Erosion. A mathematical expression is this: define two sets *A* and *B* in Z^2, the erosion of *A* by *B*, denoted $A \ominus B$, is:

$$A \ominus B = \{z|(B)_z \subseteq A\}$$

B subscript *z* means shift *B* by *z*. Similarly, the dilation of *A* by *B* is denoted as $A \oplus B$:

$$A \oplus B = \{z|(B')_z \cap A \neq \emptyset\}$$

B' is *B* reflected about its origin. This function implies that the dilation of *A* by *B* is the set of shifts *z,* such that, at least one element of $(B')_z$ is contained in *A*.

Generally speaking, erosion functions to remove noise and weak joints, whereas dilation is wildly used to fill in holes and breaks. Operation of dilation usually follows erosion, to make sure that the shape to be processed is not enlarged or shrunk. This combination of processing is called *opening*. The same combination of operations in the other order around is called *closing*. Opening and closing keep the original size of shapes.

In this project, the SE chosen is a round-like shape, size adjustable. Since surface image in Fig. 3-4d (shown in white) is contaminated by part of the head stamp (in black), thus operation of erosion is first applied (equivalent as to say the head stamp is dilated). Result is shown in Fig. 3-4a. Though inside the head profile there

are still white noises, its outline is totally connected. Thus the head profile can be completely separated as shown in Fig. 3-6, in red oval. Apply connected component (introduced in *Section 2.1*), the head profile is extracted as shown in Fig. 3-6c and Fig. 3-6d.

a. Connected outline

b. Connected head profile

c. Isolated head profile image

d. Background image

Fig. 3-6 Head profile segmentation

Transfer the head profile into its binary image (head profile in black and the rest in white) and convert it further into polar coordinates. The result is shown in Fig. 3-

7. If we define **crossing pixels** as white pixels in column whose neighbors are different, then the crossing pixel in each column is a contour point at that angle. Thus elements in the feature vector *a* are positions of crossing pixels in each column (360 columns in total).

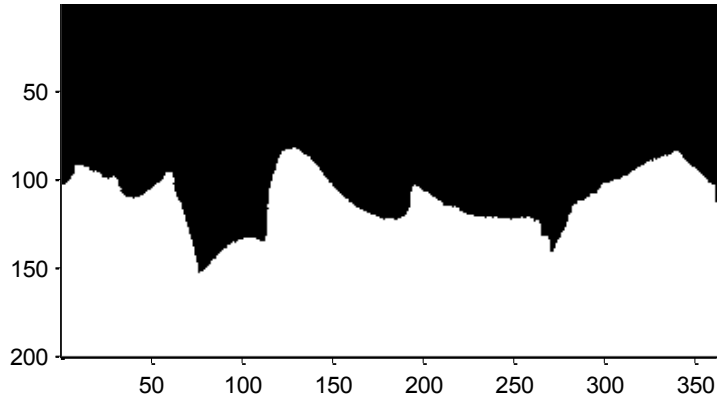
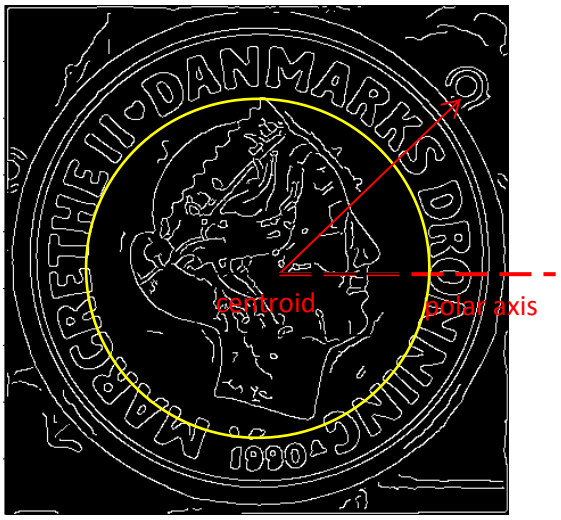


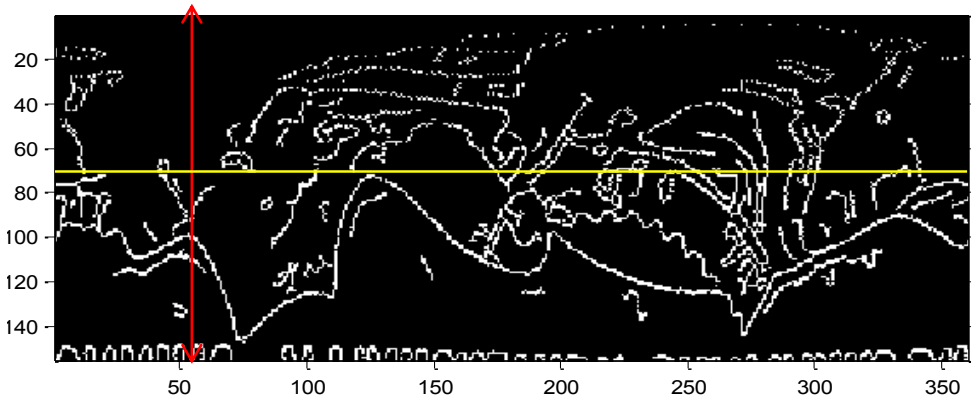
Fig. 3-7 Binary image of head profile under polar coordinates (Source: *90_ref_a*)

3.2.2 Contour Tracking

The other contour extraction algorithm is to detect edges and track down eligible edges and record it as contour. In this project, *Canny Edge* detection is applied and the result is shown in Fig. 3-8a. Its image converted into polar coordinates is shown in Fig. 3-8b. Here are some correspondences between the two images: pixels from the same column in Fig. 3-8b, as shown in red arrows, are originally lying along the same line radiated from the centroid; pixels that share the same distance from the centroids, as the yellow circle shown in Fig. 3-8a, correspond to the horizontal line in Fig. 3-8b.



a. Edge image



b. Edge image in polar system

Fig. 3-8. Correspondences of edge image

As shown in Fig. 3-8b, though the contour edge is obvious and seems extracted, there are quite a few noises and broken points distributed along and around it. Thus it is necessary to develop a good algorithm to extract the contour.

The method we propose in this project is summarized in Fig. 3-9, a flow chart to track and record the contour. The notations used in Fig. 3-9 defined as follows.

1. **Break point**: a black pixel surrounded by white pixel neighbors.
2. **Ref**: ref is a reference position of a contour point at a certain angle.

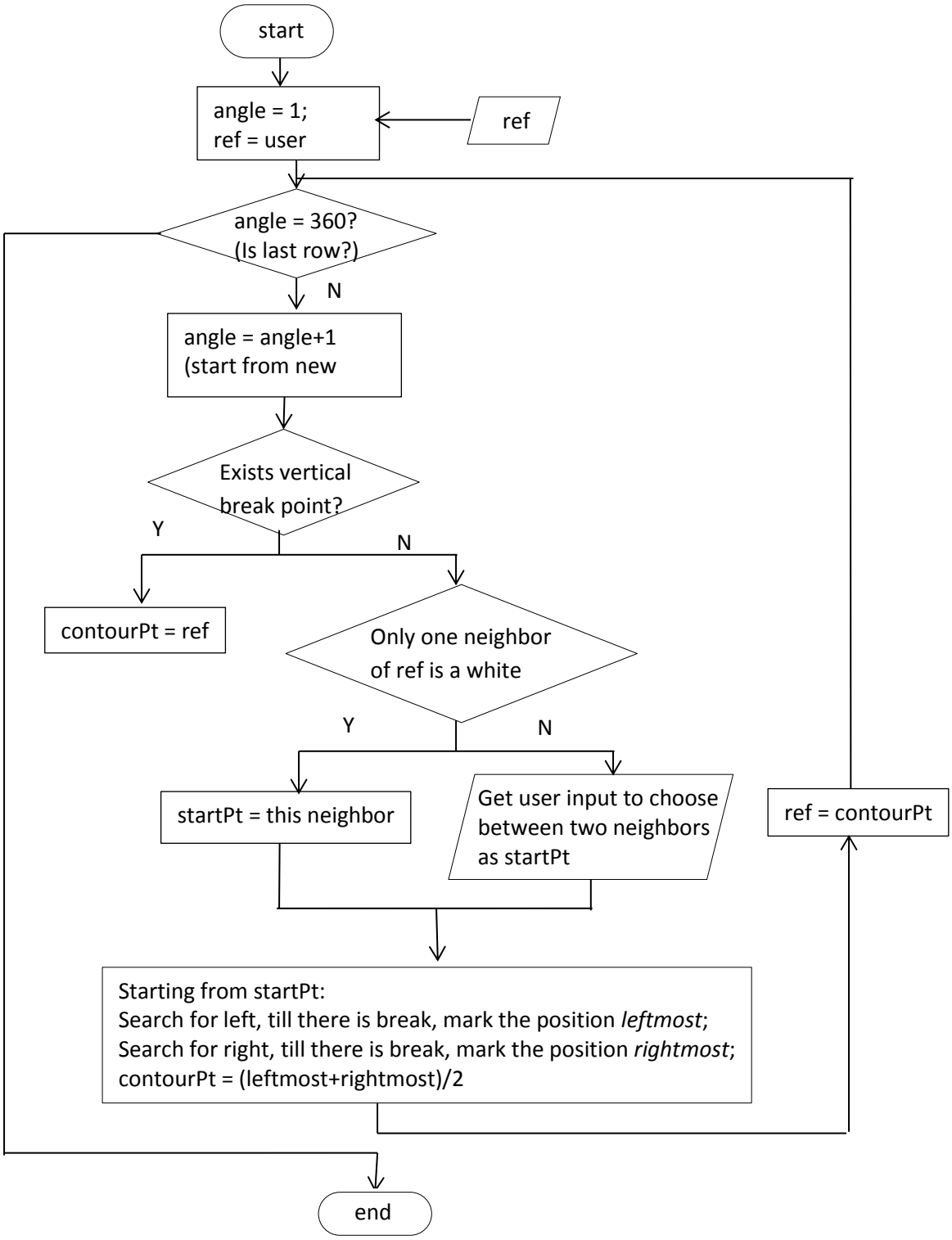
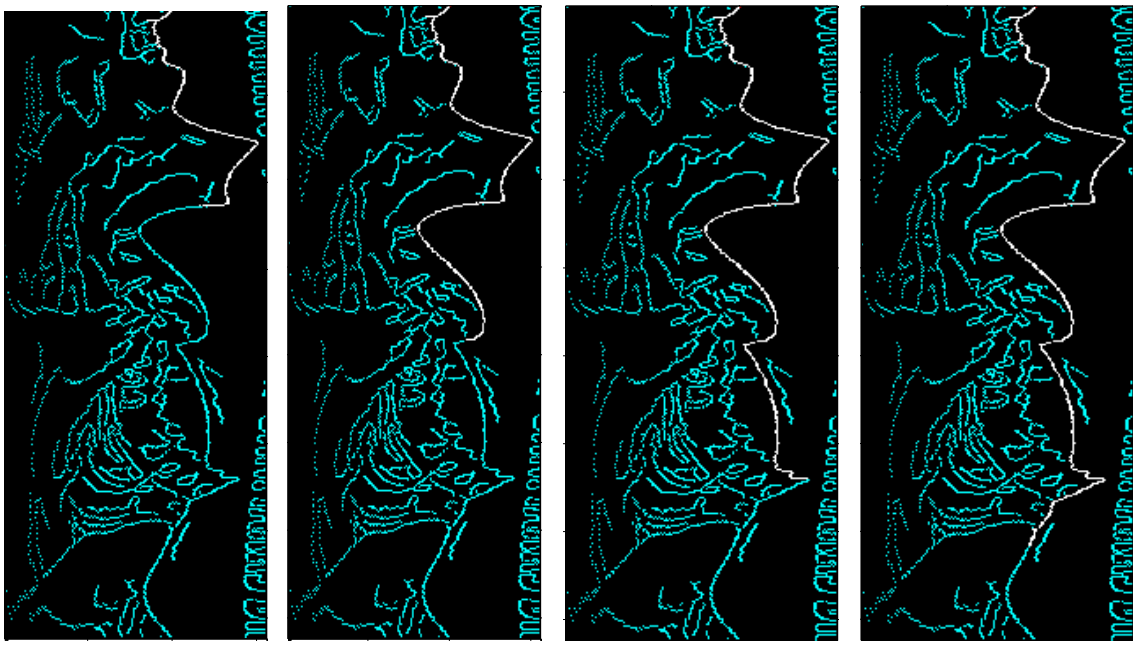


Fig. 3-9 Contour tracking flowchart

Note that Fig. 3-9 only illustrates the basic idea of this program. In practice, since there exist many break points and noises, a lot more conditions are applied, and user inputs are required occasionally. An example of practice is shown in Fig. 3-10. Fig. 3-10a to d are four phases extracted during the process.



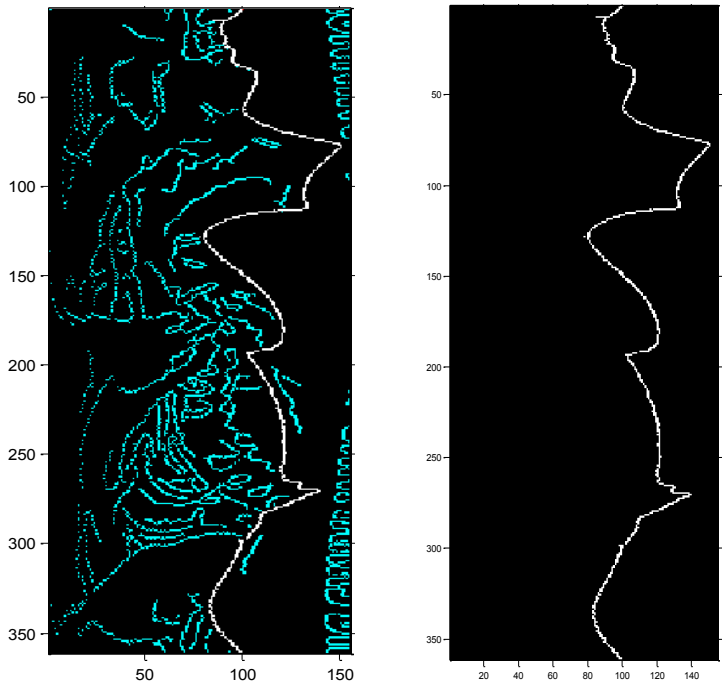
a. Phase 1 **b.** Phase 2 **c.** Phase 3 **d.** Phase4

Fig. 3-10 Contour tracking program

Fig. 3-11 is an example that shows the contour tracking result of Fig. 3-8. Pixels in blue are the original edge pixels; pixels in white are contour points. It can be observed that for each row, only one or a few pixels are selected as contour points. For convenience, image with only contour points as Fig.3-11b is recorded as ***I(σ, θ)***.

Since in "*angle-distance*" only one distance value σ is taken for each angle θ, an average of σ is calculated via Eq. 3-9 when two or more contour points lie on a row.

$$a(\theta) = \frac{\sum_{\sigma} I(\sigma,\theta)\cdot\sigma}{\sum_{\sigma} I(\sigma,\theta)} \qquad 3\text{-}9$$



a. Contour imposed on polar image **b.** Intact contour

Fig. 3-11 Contour tracking result

3.3 Experiment

In this project, we have tested sixteen coins all together. The contour extraction algorithm adopted is *foreground separating*. Within all the testing samples, eight coins are from the year 1990, and the other eight are from year 1996. Coins of the same year share one head profile pattern. The eight coins from 1990 are real; out of all the coins from year 1996, three are known real (*96_ref_a, 96_dr suen_1, 96_ref_real)*, and two are known fake (*KT3, KT5*). Besides, there are three coins (*96_fake_01, 96_fake_02, 96_fake_05*) marked as highly possibly fake due to poor qualities, *i.e.* coarse surface. These are called suspicious coins in the remaining section.

Table 3-1 records the dissimilarity measurement of each pair of the the eight coins from year 1990. In each tuple *(coin1, coin2)*, the value on the top represents

dissimilarity measured using Eq. 3-7, and on the bottom is the misaligned angle of *coin1* to *coin2,* measured by Eq. 3-8. Note that in this table, the dissimilarity in tuples is diagonal symmetric and thus is only recorded once. (The similarity values are not shown in the lower left triangle, since they are duplicates from their upper triangle counterpart.) Also note that the misaligned angle is diagonal symmetric too, but their signs are mutually negative.

Table 3-1. Shape Dissimilarity between Coins -Year 1990

	90_Dr Suen_1	**90_Dr Suen_2**	**90_Dr Suen_3**	**90_ref_a**	**90_ref_b**	**90_ref_c**	**90_ref_d**	**90_ref_e**
90_Dr Suen_1		9.15 -5	3.75 -4	5.19 -7	2.34 -3	2.98 0	2.74 0	5.93 3
90_Dr Suen_2	5		9.712 2	6.179 -3	9.711 2	9.109 5	8.422 5	8.276 6
90_Dr Suen_3	4	-2		7.237 -4	4.845 2	3.193 4	5.271 4	5.525 6
90_ref_a	7	3	4		5.080 4	6.043 7	3.943 7	6.759 8
90_ref_b	3	-2	-2	-4		3.833 4	2.185 4	6.173 5
90_ref_c	0	-5	-4	-7	-4		3.837 0	4.122 3
90_ref_d	0	-5	-4	-7	-4	0		5.535 2
90_ref_e	-3	-6	-6	-8	-5	-3	-2	

Two conclusions can be drawn from Table 3-1:

The differences between real coins are very small. Normally, on average there is only a difference of 3 to 7 pixels with regarding to σ (σ on average is 200 pixels);

Table 3-2 summarizes the cross validation results from coins of year 1996. Note that for fake coins – *KT3 and KT5*, the head profile is not extractable due to the poor depth design, thus by default, they are set as the average σ of contour points of its comparison coin.

Table 3-2. Shape Dissimilarity between Coins -Year 1996

	96_ref_a	**96_drsuen_1**	**96_ref_real**	**96_fake_01**	**96_fake_02**	**96_fake_05**	**KT3**	**KT5**
96_ref_a		4.9983	5.140-3	7.6660	3.4544	4.041-2	106.57	106.57
96_drsuen_1	-3		6.179-3	9.7112	5.2552	4.489-4	106.99	106.99
96_ref_real	3	3		8.5654	5.6006	4.6002	106.52	106.52
96_fake_01	0	-2	-4		7.2224	7.596-3	109.36	109.36
96_fake_02	-4	-2	-6	-4		5.103-5	107.57	107.57
96_fake_05	2	4	-2	3	5		107.25	107.25

This algorithm verifies the fake coins, *KT3 and KT5*, to be fake; meanwhile for the suspicious coins, *96_fake_01, 96_fake_02, 96_fake_05*, in terms of their head profile shapes, these are probably genuine coins. Further tests are required to give more proofs for its authenticity.

Fig. 3-12 provides a straightforward evidence of the results. Both figures show the merged result from two coin images. In Fig. 3-12a, it is a template real coin (*96_ref_a*) in red channel and a fake coin (*KT3*) in green channel, whereas in Fig. 3-12b, it is the same template coin in red channel and a suspicious coin (*96_fake _01*) in green channel. If two images merge well, the color is yellow. Comparing the results, it is obvious that the two images in Fig. 3-12a are very different: the head profile of *KT3* is larger and the lines around the chin area are totally varied. However, though claimed as suspicious, *96_fake_01* is merging very well to the template coin. This coincides with our experimental results.

a. *KT3* and *96_ref_a* **b.** *96_fake_01* and *96_ref_a*

Fig. 3-12 Image merging comparison

3.4 Conclusion

In this chapter, we have designed a new feature to describe and encode shapes. It is a solid algorithm since it is representative and also computational efficient due to its rotational invariance property. This feature is applied to detect the counterfeit coins. To accomplish that, two algorithms have been designed for coin images, foreground separating, and contour tracking. Finally, we carried out experiments on two image patterns of coins, and cross-validated the results. For the coins of year 1990, all the coins are real. The results showed very small dissimilarities, only 2 to 7 pixels in difference. For the other coins of year 1996, three coins are real, three are suspicious, and two are fake ones. The results showed high similarities among the real coins and the suspicious ones. On the other hand, for the good quality yet truly fake coins (*KT3, KT5*), their images are not extractable in one whole piece. A big discrepancy occurs with about 105 pixels. Thus they can be recognized easily as counterfeit coins.

Chapter 4 Texture Analysis

4.1 Texture Analysis Review

Apart from letterings and images, the other accessible visual property on the coins surface is texture. Though texture is hardly officially defined, it can introduce quite a lot of information about objects, such as roughness, density, uniformity, smoothness, granulation, *etc*.

According to Materka [38], texture analysis can serve four purposes:

1. Extract features to numerically represent texture properties;
2. Distinguish combined texture patterns into homogenous sections;
3. Classify texture and determine which pattern it resembles;
4. Extract texture information to assist 3D reconstruction.

The essences of goals from 1 to 3 are similar, *i.e.* texture representation and its applications in classification. The implementations of each subsequent goal are based on the success of its previous goals. Numerous mature algorithms have been proposed to that end. Generally speaking, the methods can be divided into 3 categories:

- **Statistical approaches**: statistical descriptors or features are designed to represent distributions of pixel intensity;
- **Structural approaches:** better used for synthesis, structural approaches have a set of predefined shapes, called primitives, and rules, based on which new texture can be generated;
- **Spectral approaches:** Fourier transform, Gabor transform, and wavelet transform, *etc*. The information to be seeking is distinguishing frequency and size.

By comparison, the last application is a bit different. Instead of trying to find numerical descriptors of the texture, it focuses on structures, local features or even points, which are discriminant, invariant and stable [39]. Specifically, it means those regions can be detected with high repeatability in different images of the same scenes or the same objects. These structures are often called distinct regions (DR) [40], invariant regions [41] or covariant regions [42], which provide a compact and robust representation of patterns in an image. The detection of DR now is widely used in applications where finding correspondences is heavily involved, for instance, object recognition, stereo matching, object tracking, robot navigation *etc*.

Tuytelaars, T. et. al. have studied different kinds of DR features [39], how they evolved and applied. Among the many DR features, there is a particular group called affine invariant feature descriptor. Affine invariant features are descriptors that are consistent with images of the same scene yet taken from different angles. Regardless of the appearance of the surroundings of the actual region, those features stay unchanged [43]. In this section, we choose and investigate in one of the affine invariant features – Maximally Stable Extremal Regions (MSER), and study how it can contribute to the texture analysis of coins.

4.1.1 Maximally Stable Extremal Regions

MSER is first defined by *Matas* in gray scale images [44]. According to Matas, extremal regions have two desirable properties: the set of regions is closed under perspective transformation, and the set is closed under monotonic intensity transformation. On top of that, MSER also possesses an extremal property of intensities within the region and along the boundary. That is, MSERs are homogenous regions or areas that possess high intensity contrast to their

surroundings, either brighter or darker, and are stable across a range of threshold changes. MSERs have also been extended on other scalar functions [45], multi-resolution MSER [43], color MSER [46] and efficient MSER [47].

The idea of MSER was well explained by a metaphor according to *Matas*. If a movie of thresholded image I_t, with t referring to threshold value, what is shown with *t* accumulating is black spots corresponding to local intensity minima which appear and grow, till all is black eventually; or the other way around, with t decreasing, more white spots are appearing and growing till it becomes all white. The set of all connected components across a wide range of *t* is the set of extremal regions. In the first scenario, the extremal regions are called maximal regions, or MSER+, which are bright homogeneous areas with darker boundaries; on the contrary, the regions in the second scenario are called minimal regions, or MSER-, which are dark areas with brighter boundaries. In both cases, the union of sets is used as MSER detection result.

A more formal definition of MSER is summarized below.

MSER Definition

To formally define MSER, a few relative concepts are necessary.

Region: *R* is a contiguous subset of the original image. That is, for each *p, q*<*R*, there is a path within *R*, connecting *p* and *q*.

Boundary: *B(R)* is a subset of pixels that are adjacent to at least one pixel from *R*, yet do not belong to *R*;

Extremal Region: R_i^g is a region that $\forall p \in R, \forall q \in B(R), I(p) > I(q)$ (MSER+) or $I(p) < I(q)$ (MSER-). $I(\cdot)$ is the intensity of pixel. The superscript g is a

threshold value applied to the original image; whereas subscript *i* is the index of the regions;

MSER: for a sequence of threshold value $g \in [g_0, g_0 + \Delta, g_0 + 2 * \Delta, \dots, g_0 + (n-1) * \Delta]$ $(g_0 > 0, g_0 + n * \Delta < 256)$, if $\Psi(R_j^g) = (|R_j^{g+\Delta}| - |R_j^g|)/|R_j^g|) < \mu$, then it is a local extremal. Δ and μ are parameters of this method.

4.2 Hole Detection

Holes or other types of indents are straightforward indicators of coin qualities. And since these are areas relatively darker than their surroundings, it fits perfectly with the definition of MSER-. Thus in this section, we deployed MSER- detector to locate holes and indents on the image surface. The results can be used as part of the references to a coin quality. Generally speaking, the more holes or indents are spotted, the lower the quality is graded.

The algorithm adopted in this project is shown in Fig. 4-1. Considering the computational efficiency, we adopt an idea that is closer to the implementation proposed by *Donoser, M., et al*. [47]. g_0 is the user-defined lowest threshold value of intensity, *n* is user-defined iteration time, and Δ is the user-defined intensity interval. Both g_0 and Δ are positive, and (g_0, ($g_0 + n * \Delta$)) is bounded within 0 to 255. Each time, the threshold value increases by Δ, and images thresholded by two consecutive intensity values $(g_0 + i * \Delta)$ and $(g_0 + (i+1) * \Delta), i \geq 0$ are compared. Only those filtered regions whose sizes from two consecutive results have shrunk smaller than μ are kept, and are processed as potential holes/indents. The union-find data structure described in [47] is designed to detect connected components, same as blob detection depicted in Fig. 4-1; and the tree structure is helpful to iterate and update new extremal areas.

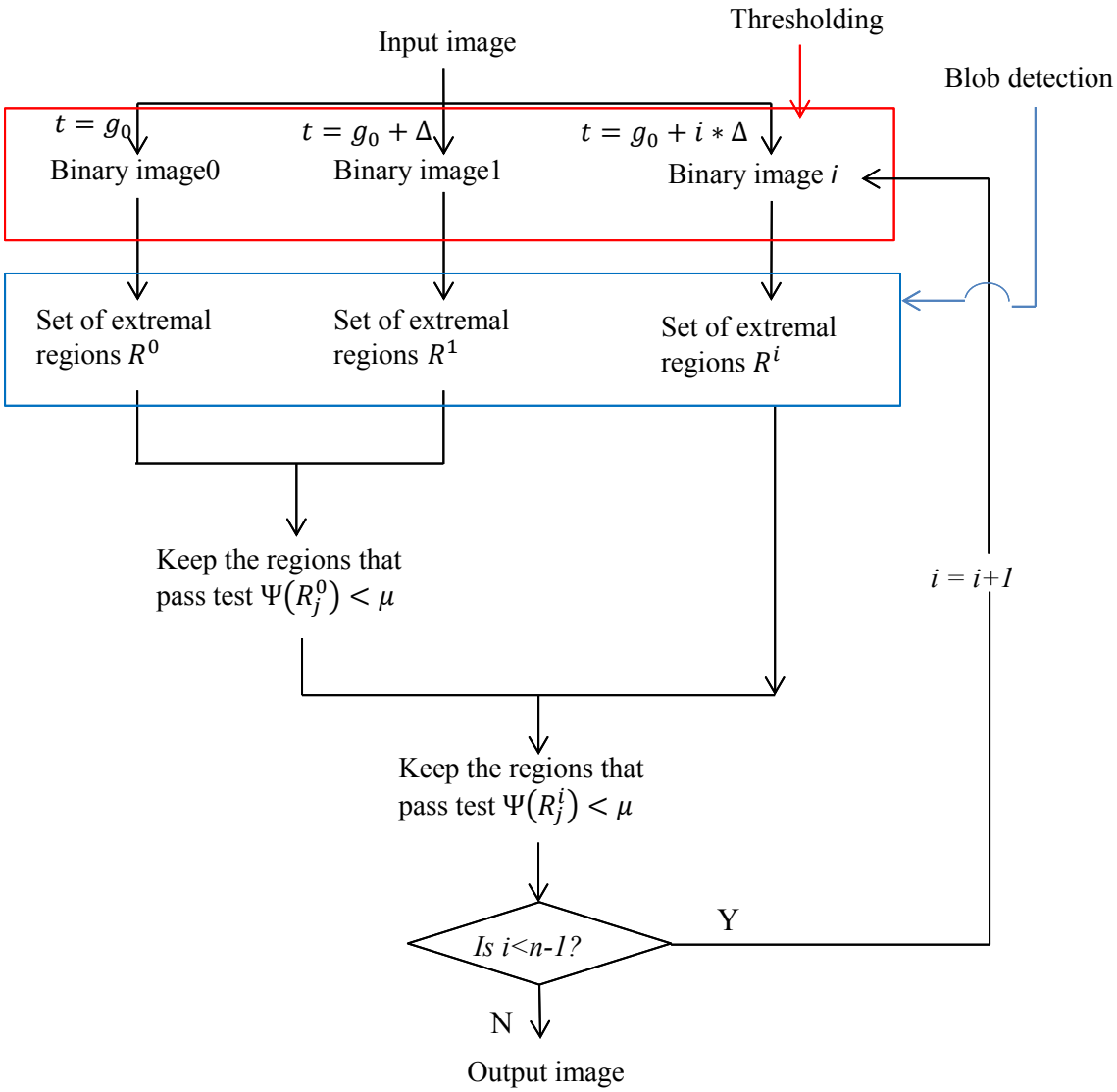


Fig. 4-1 Program of MSER

4.3 Experiment

Three experiments are conducted in this section.

First, we shall test the best thresholding range, i.e. $(g_0, g_0 + n * \Delta)$, based on the performance of the detection of holes.

Second, we shall test the best value of μ values, to filter potential holes/indents.

Third, we test the setting of a combination, μ and Δ.

Last, based on the parameters chosen from the previous two steps, we will try MSER- described in section 4.2 on eight coins, and record the number of holes.

4.3.1 Configuration of Extension over $(g_0, g_0 + n * \Delta)$.

In this section, the coin *90_Dr Suen_2* is adopted for instance, as shown in Fig. 4-2. The objective is to detect the holes/indents as marked in read ovals. To that end, we have tested three combinations of threshold range $(n * \Delta)$ and interval value Δ. By different configurations, the results are checked, and the effectiveness is evaluated based on human judgment.

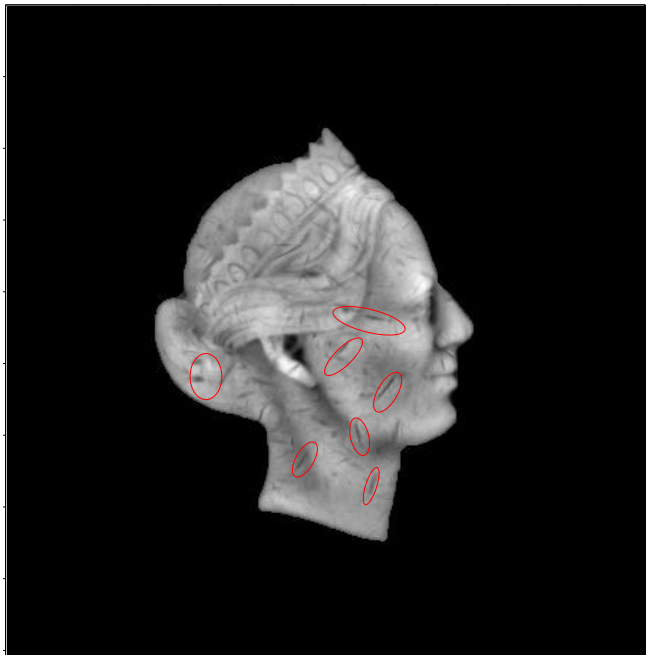
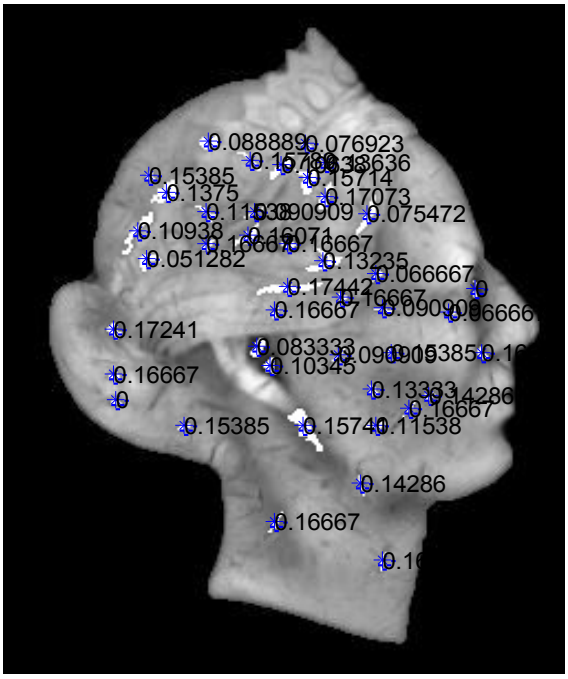


Fig. 4-2 Original image with holes marked in red (Source: *90_Dr Suen_2*)

Below in Fig. 4-3 are the detected results. The configurations from figures *a* to *c* are (20, 20+3*45), (20, 20+5*27), (20, 20+3*27) respectively. Areas shown in white are detected holes under that configuration, and the digits marked beside

them are ψ, the change ratio. In this section, the value of ψ maximally can be 0.175. For the rest of this chapter, figures are depicted in the same manner.



a. Conf. 1: (20, 20+3*45)

b. Conf. 2: (20, 20+5*27) **c.** Conf. 3: (20, 20+3*27)

Fig. 4-3 Detection results with different extensions

Table 4-1 records all the detected areas, including their position and value ψ. *Conf.1* to *conf.3* correspond to Figs. 4-3a to 4-3c.

Table 4-1 Records of Holes Detected with Different Extensions

Index	Coordinates of detected centroid holes		ψ		
			conf. 1	*conf.* 2	*conf.* 3
1	203.6	243.6	0.103	0.150	0.103
2	198.2	235.1	0.083	0.143	0.083
3	296.6	208.2	0.000	0.164	0.000
4	299.9	238.1	0.167	0.171	0.167
5	219.4	272.4	0.157		0.157
6	212.2	186.9	0.167	0.154	0.167
7	266.8	263.7	0.167	0.083	0.167
8	206.2	316.7	0.167		0.167
9	132.9	227.3	0.172		0.172
10	245.1	299.4	0.143	0.143	0.143
11	255.1	217.5	0.091	0.125	
12	197.0	171.6	0.091	0.077	
13	252.1	201.0	0.067		
14	275.7	257.6	0.143		
15	133.7	260.3	0.000	0.154	
16	193.9	183.5	0.161		
17	236.3	211.5	0.167		
18	254.6	335.0	0.167		
19	175.4	187.2	0.167		
20	234.7	238.5	0.091	0.154	
21	250.3	255.2	0.133		
22	175.3	172.5	0.115		
23	143.5	180.8	0.109		
24	164.7	272.0	0.154		
25	228.5	165.0	0.171		
26	149.3	154.9	0.154		
27	133.1	248.3	0.167	0.133	
28	206.1	218.0	0.167	0.063	
29	247.8	173.3	0.075		
30	251.6	271.5	0.115		
31	258.6	237.9	0.154		
32	156.6	162.8	0.138		
33	220.0	140.4	0.077		
34	228.2	195.0	0.132		
35	176.2	139.0	0.089		
36	211.6	207.4	0.174		
37	208.7	150.4	0.106		
38	220.8	156.4	0.157		
39	285.3	218.5	0.067	0.067	
40	148.3	194.4	0.051	0.167	
41	229.2	149.1	0.136		
42	194.8	147.7	0.158		
sum			42	15	10

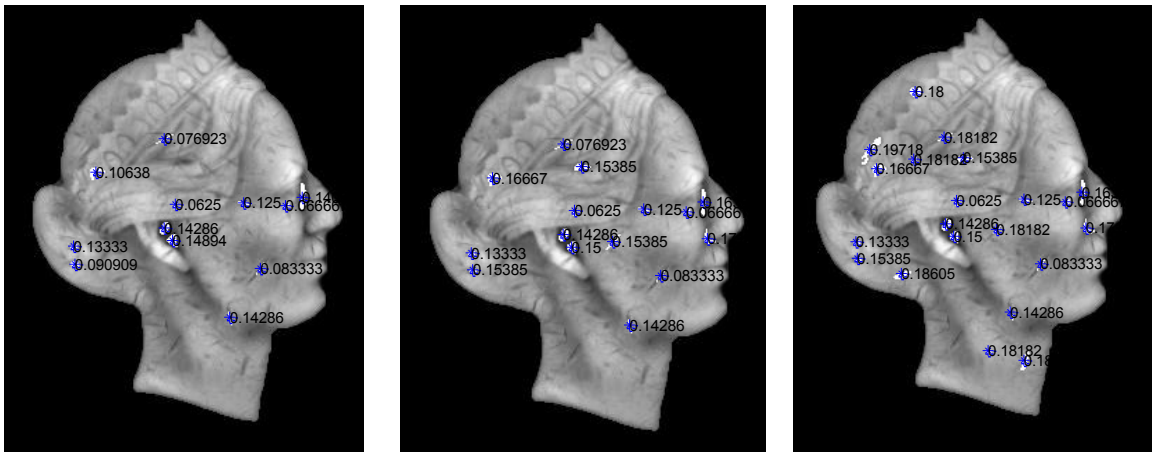
Several conclusions can be drawn from Table 4-1.

1. Comparing the results from *conf.1* and *conf.3,* out of the 42 holes detected by the former, results from *conf. 3* only consist a part, and are biasedly distributed. The difference is due to an extension of threshold value: for *conf.1*, the value is more widely spread, which is (20+3*45)-20=135, whereas for the latter, it is (20+3*27)-20=81. We think a larger extension like *conf.1* outperforms smaller extension since they will output more inclusive results. Also note that though technically, the extension could be as large as 0 to 255, it is not practical since intensity values of pixels in this example are distributed in a narrower range around 20 to 160. Thus, the biggest range we shall use in the latter part would be 20 to 150.

2. Comparing the results from *conf.1* and *conf.2,* the extensions are identical, yet the interval value Δ is larger of *conf.2*. Δ suggests sensitiveness: generally speaking, the larger Δ is, the fewer holes will be detected. This does not imply that a smaller Δ is always better though. The potential negative effects with a small Δ are falsely detected holes. Thus it is a trade-off between inclusiveness and over-sensitiveness. Also, note that the distribution in *conf.2* is even, which is a good sign for not being biased. Meantime, the choice of Δ is also affected by μ, as explained in section 4.3.3.

4.3.2 Configuration of Threshold ψ , μ

In this section, we shall still use the coin *90_Dr Suen_2* as an example, as shown in Fig. 4-2. The range of thresholds is (20, 20+5*26). The objective is to choose the best μ, which is a threshold value for ψ. The three values of μ chosen in this section are respectively 0.125, 0.175 and 0.200, corresponding to Fig. 4-4a to Fig. 4-4c.



a. $\mu = 0.125$ **b.** $\mu = 0.175$ **c.** $\mu = 0.200$

Fig. 4-4 Detection results with different μ values

Table 4-2 shows records of the detected holes, including their position and value ψ. *Conf.1* to *conf.3* correspond to Figs. 4-4a to 4-4c.

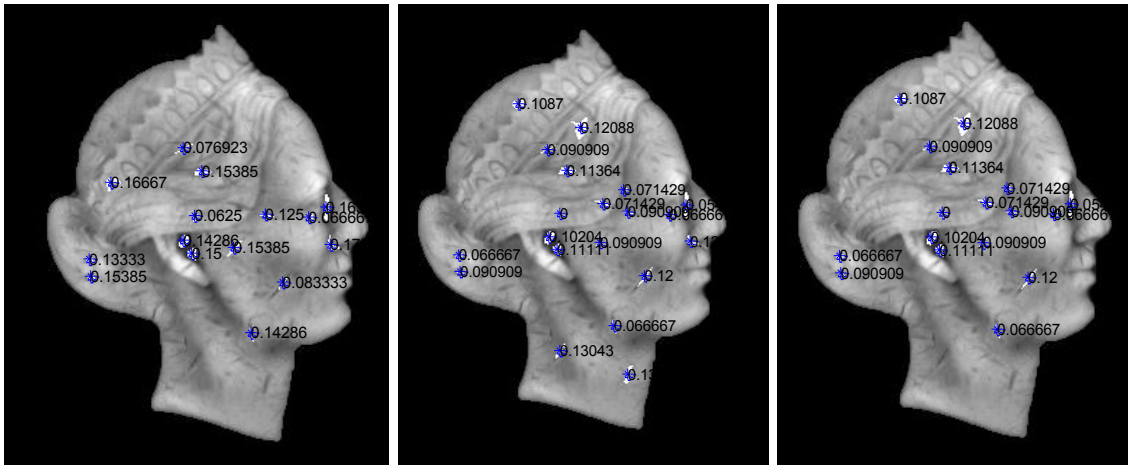
Table 4-2 Records of Holes Detected with Different μ Values

Index	Coordinates of detected holes centroid		ψ		
			conf. 1	*conf. 2*	*conf. 3*
1	198.2	235.0	0.143	0.143	0.143
2	204.0	243.7	0.150	0.150	0.149
3	296.9	212.2	0.164	0.164	0.141
4	300.3	237.8	0.171	0.171	
5	266.8	263.7	0.083	0.083	0.083
6	210.8	187.0	0.154	0.154	
7	245.1	299.4	0.143	0.143	0.143
8	197.0	171.6	0.182	0.077	0.077
9	174.6	187.8	0.182	0.125	
10	255.0	217.3	0.125	0.154	0.125
11	134.0	260.2	0.154	0.154	0.091
12	234.7	238.5	0.182		
13	143.4	180.8	0.197		
14	254.9	333.9	0.186		
15	133.1	248.2	0.133	0.133	0.133
16	166.0	271.4	0.186		
17	176.0	138.7	0.180		
18	205.9	217.9	0.063	0.063	0.063
19	148.4	194.6	0.167	0.167	0.106
20	285.3	218.5	0.067	0.067	0.067
21	228.6	326.9	0.182		
sum			21	15	12

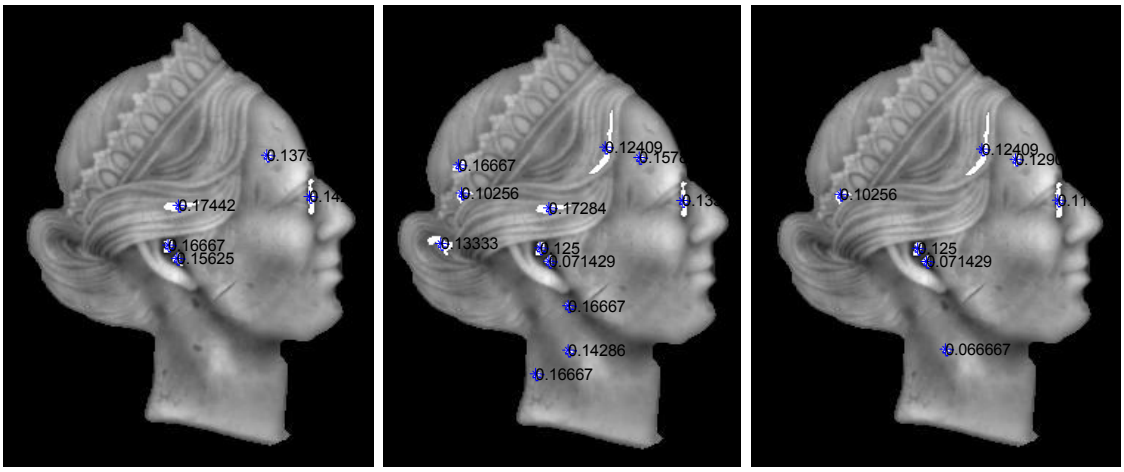
From the results, we can tell that the larger μ is, the more holes will be detected. However, given some holes are falsely detected, we shall use a reasonable value of μ that gives the best trade-off. Also, the choice of μ is pertained to the choice of Δ. Details are elaborated in the next section.

4.3.3 Configuration over μ and Δ

From the pervious two sections, it is acknowledged that the smaller Δ is, or the larger μ is, the more sensitive the detection is. Another rule to be revealed in this section is that the smaller Δ is, the smaller μ will be. Below we adopted two coin images, *90_Dr Suen_2*, and *90_ref_e*. The set-ups for the first example are shown in Fig. 4-5, from *a* to *c*, the parameters are respectively (20,20+5*26), $\mu = 0.175$, (20,20+4*32), $\mu = 0.132$, and (20,20+4*32), $\mu = 0.125$; the set-ups for the second coin are shown in Fig. 4-5, from *d* to *f*. The parameters are (20,20+5*26), $\mu = 0.175$, (20,20+4*32), $\mu = 0.132$, and (20,20+4*32), $\mu = 0.125$.



a. (20,20+5*26), $\mu = 0.175$ **b.** (20,20+4*32), $\mu = 0.132$ **c.** (20,20+4*32), $\mu = 0.125$



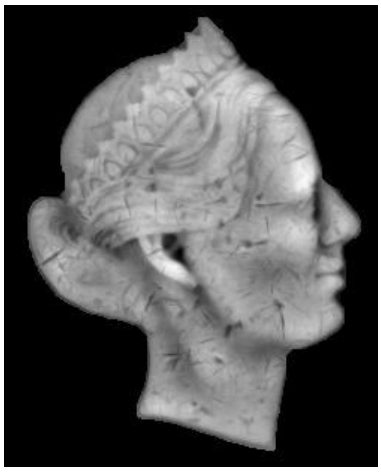
d. (20,20+5*26), $\mu = 0.175$ **e.** (20,20+4*32), $\mu = 0.132$ **f.** (20,20+4*32), $\mu = 0.125$

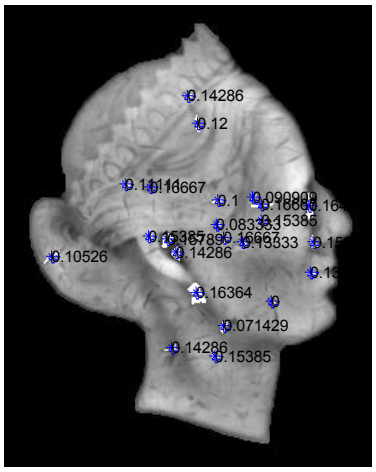
Fig. 4-5 Detection results with different μ and Δ values (Source: first row- *90_Dr Suen_2*, second row: *90_ref_e*)

Comparing the results of Figs. 4-5b and c, or Figs. 4-5e and f, with a smaller μ, part of the detected results is cut off. However, the differences between Figs. 4-5a and 4-5c, Figs. 4-5d and 4-5f are not notable. Images in the middle for both coins are more inclusive, but it also mistakenly detects unexpected parts in the hair. As a result, we use the first configuration (20,20+5*26), $\mu = 0.175$ as a standard, and all the coins tested and compared in next section shall adopt this set-up.

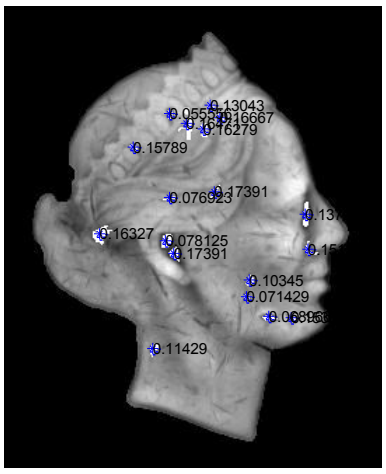
4.3.4 Experimental Results

In this section, we have tested eight coins: they are *90_Dr Suen_1, 90_Dr Suen_2, 90_Dr Suen_3, 90_ref_a, 90_ref_b, 90_ref_c, 90_ref_d and 90_ref_e* respectively. The configuration as mentioned in section 4.3.3 is (20,20+5*26), $\mu = 0.175$. Beside the two coins are already shown in Fig. 4-5, the results of the remaining are shown in Fig. 4-6.

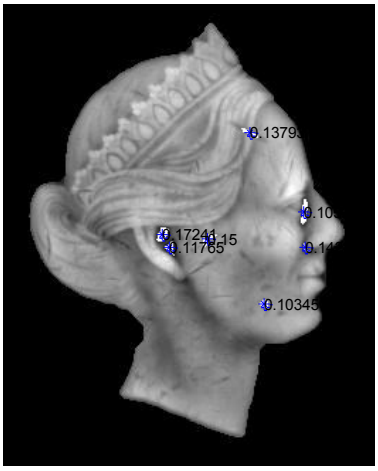




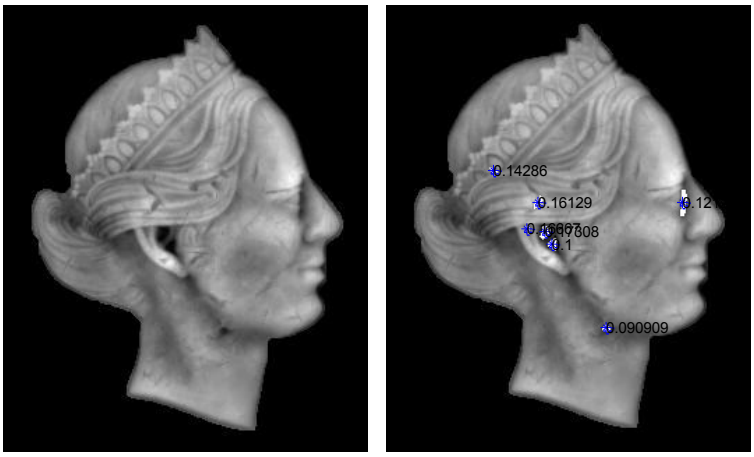
a. Original image (source: *90_Dr Suen_1*) **b.** Detected result of a



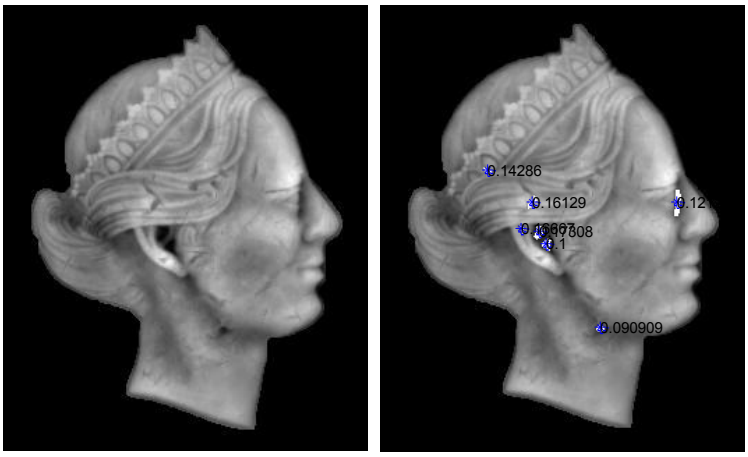
c. Original image (source: *90_Dr Suen_3*) **d.** Detected result of c



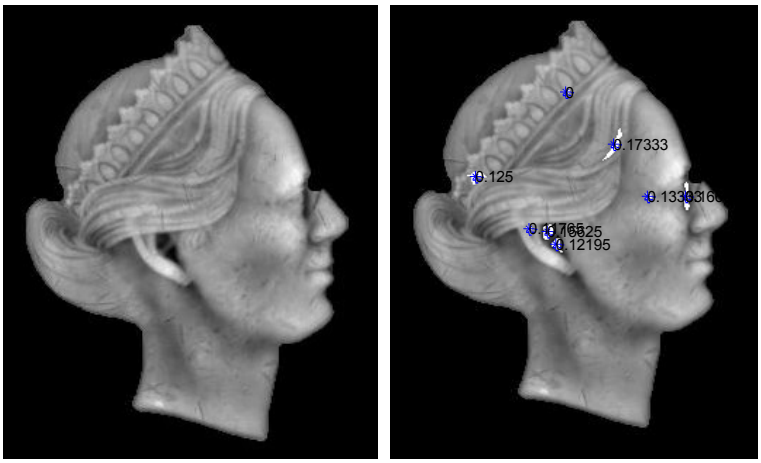
e. Original image (source: *90_ref_a*) **f.** Detected result of e



g. Original image (source: *90_ref_b*) **h.** Detected result of g



i. Original image (source: *90_ref_c*) **j.** Detected result of h



k. Original image (source: *90_ref_d*) **l.** Detected result of j

Fig. 4-6 Detected results with configuration: (20,20+5*26), $\mu = 0.175$

Table 4-3 Detected results with configuration: (20,20+5*26), $\mu = 0.175$

Coordinates of centroids of the detected holes		ψ
90_Dr Suen_1		
210.0	238.2	0.143
203.8	229.3	0.158
302.2	206.3	0.164
207.2	304.3	0.143
242.8	288.9	0.071
305.2	231.5	0.156
224.1	266.3	0.164
192.4	193.2	0.167
123.7	241.1	0.105
276.4	272.0	0.000
256.1	231.3	0.133
263.4	199.7	0.091
269.0	205.0	0.167
238.3	218.5	0.083
303.1	252.3	0.133
218.5	129.6	0.143
236.7	309.5	0.154
224.8	149.0	0.120
238.6	201.9	0.100
175.4	191.1	0.111
242.7	228.1	0.167
268.5	216.3	0.154
191.4	227.5	0.154
90_Dr Suen_2		
198.2	235.0	0.143
204.0	243.7	0.150
296.9	212.2	0.164
300.3	237.8	0.171
266.8	263.7	0.083
210.8	187.0	0.154
245.1	299.4	0.143
198.0	171.1	0.077
255.0	217.3	0.125
134.0	260.2	0.154
233.3	239.8	0.154

Coordinates of centroids of the detected holes		ψ
90_Dr Suen_3		
241.0	0.122	0.174
231.7	0.156	0.078
207.0	0.167	0.138
170.4	0.173	0.163
192.7	0.125	0.152
230.2	0.118	0.069
132.7	0.000	0.071
206.6	0.133	0.103
209.8	141.2	0.056
183.8	165.1	0.158
209.8	201.1	0.077
296.8	286.6	0.154
198.1	308.5	0.114
241.6	196.6	0.174
246.2	143.6	0.167
234.1	152.5	0.163
238.9	134.8	0.130
222.4	148.2	0.165
90_ref_a		
205.4	237.9	0.118
200.0	228.5	0.172
300.1	212.6	0.105
301.4	238.4	0.143
261.7	157.0	0.138
271.9	277.6	0.103
232.1	231.6	0.150
90_ref_b		
206.8	223.9	0.173
212.1	232.9	0.100
250.5	293.3	0.091
305.9	203.0	0.121
194.3	221.8	0.167
169.6	179.8	0.143
201.6	202.5	0.161

Coordinates of centroids of the detected holes		ψ
90_ref_c		
210.3	241.0	0.122
204.0	231.7	0.156
302.2	207.0	0.167
251.2	170.4	0.173
153.1	192.7	0.125
191.1	230.2	0.118
217.3	132.7	0.000
275.4	206.6	0.133
90_ref_d		
203.4	230.2	0.153
209.8	239.6	0.103
301.1	204.5	0.087
152.4	191.8	0.125
249.9	181.4	0.118
163.3	188.6	0.133
90_ref_e		
203.5	232.8	0.167
209.8	241.9	0.156
300.4	200.3	0.143
210.5	205.7	0.174
270.9	171.9	0.138
90_Dr Suen_2 (Continue)		
133.1	248.2	0.133
205.9	217.9	0.063
148.4	194.6	0.167
285.3	218.5	0.067

4.3.5 Conclusion

By configuring the best set-ups of MSER-, our algorithm automatically detected the holes on the surface of head profile. By comparison, the qualities of the first three coins (recorded as *90_Dr Suen_1, 90_Dr Suen_2, 90_Dr Suen_3*) are relatively low with more deep scratches, holes and indents. The results indeed reflect this observation: the number of holes detected on average is more. Thus we think it is a valid way to assess the quality of coins.

Chapter 5 Conclusion

This thesis is focused on the detection of counterfeit coins, where coins are well casted and have no obvious traits of being fake. Acknowledged that counterfeit coins are not all the same, we hope to explore various aspects from coins and find as many valid features as can be used to distinguish counterfeit and genuine coins. Obtaining a group of reliable features and being able to use them to detect fake coins would be helpful for the general public, *i.e.* reduce crime, keep stable market, and reduce individual losses.

Given that coins are relatively small in size and physically no tools can be applied directly to measure the fine details on the surface of coins, a new machine *IBIX TRAX* is deployed to scan coins and sample the images. Altogether we have proposed eight groups of features, concerning three visual aspects, namely lettering, image and texture. For different aspects, the images are processed differently, via combined methods of various image processing techniques. On top of that, a few feature extraction methods are proposed, including stroke width, smoothness, size, relative angle and distance of letterings (chapter 2); a novel shape feature *angle-distance* of head profile (chapter 3); and MSER of texture. Two machine learning techniques are deployed: they are max spacing *K*-clusteing and dissimilarity measurement. Three sets of experiments are carried out and the results show the methods can successfully differentiate counterfeit coins from real ones and provide insight into quality assessment.

5.1 Contribution

This thesis contributes to the detection of counterfeit coins by looking for features that can successfully distinguish fake coins from genuine ones. With regard to the

contributions provided by our work, a detailed explanation is summarized in the following paragraphs.

First, we have explored a combination of methods, including circle detection, binarization, blob extraction, erosion and dilation, *etc.* to successfully segment individual letterings, head profile, and the surface of coins. These have prepared the coin images for further feature study. Those results could be applied directly to other research concerning segmentations.

Second, we have proposed six groups of reliable features from letterings, and applied max spacing *K*-clustering to classify the counterfeit coins and real ones. Letterings encode a great deal of information, which usually is not visually noticeable due to their small size. However, by enlarging coins images while sampling, features get revealed. We extracted and quantized those information, including stroke width, size, smoothness, inter-distance, and inter-angles. As for stroke width, we used a novel heuristic quantization that is fast yet stable; image size is based on letter's principal axis and its minimum bounding box, as is inter-angle; smoothness is the ratio of area to contour length, which reflects the sharpness of contours; and inter-distance detects possible dislocation of the letterings. All such features can be applied directly to coin study, or research on letterings.

Third, we proposed a novel shape feature, *angle-distance*. This feature has three decent properties: first, it is representative that can be used for shape reconstruction; second, it is sensitive to catch small changes in shapes; third, it is rotational invariant, *i.e.* misalignment in rotation angles can be tackled as a cyclical shift. Validations of those properties are proved in chapter 3. We have also adopted a

dissimilarity measurement that not only can be used for shape comparison, but also for rotational alignment.

Then, we have applied an affine invariant feature MSER for texture analysis. MSER has widely been used in applications of object recognition, yet never in texture analysis. In this thesis, we demonstrated how it can be applied to automatically detect holes and indents on the surface of coins. Also we showed how the adjustments of input parameters can affect the detecting results.

Last but not the least, the whole research can be used for real-world detection of counterfeit coin, as is our original goal. The methods mentioned in this thesis can also help other research with similar objectives.

5.2 Future work

As stated at the beginning of this thesis, fake coins from different sources shall have different fake traits; there is yet no single feature that can be applied to tell all fake coins from real ones. Thus in this thesis, we tried working on different aspects of coins, in order to get as various features as to be thorough for coin inspection. However, it is far from being all-inclusive, and for each of the categories more features can be explored.

First of all, there is big room for research on images. In this thesis, our focus of the images is mainly on the comparison of contours, not much work has been done on the design of the image. By separating different layers of the images, more insight on the design can be revealed.

Second of all, as for texture study, more features can be designed and tested. A large proportion of earlier texture analyzing works are focusing on statistical, structural or spectral approaches. Though some proved effective, they are

generalized features. With the advancing 3D modeling and reconstruction techniques, DR/SIFT features are becoming more popular. Those features can provide new perspectives on texture study.

Last but not the least, visual aspects other than lettering, texture and image can be inspected, color, for instance. It will give a more thorough result to our study.

References:

[1] Goldsborough, R. Counterfeit Coin Detection. http://rg.ancients.info/guide/counterfeits.html, 2013.

[2] Inksure Technologies. Coin Anti-Counterfeiting. http://www.inksure.com/banknotesecurity/254-coin-anti-counterfeiting, 2014.

[3] The Royal Mint, UK. £1 Counterfeit Coins. http://www.royalmint.com/discover/uk-coins/counterfeit-one-pound-coins.

[4] Australian Federal Police. Policing: Counterfeit Currency. http://www.afp.gov.au/policing/counterfeit-currency.

[5] Wikipedia. Currency Detector. https://en.wikipedia.org/wiki/Currency_detector.

[6] Ultra Electronics Forensic Technology Ltd. IBIS BULLETTRAX-3D: The Only System to Automatically Image & Compare 3D images of Fired Bullets, 2009.

[7] Ultra Electronics Forensic Technology Ltd. IBIS BULLETTRAX-3D: Benefits of Combining 2D and 3D Image, 2009.

[8] Sun, K., Feng, B.-Y., Atighechian, P., Levesque, S., Sinnott, B. & Suen, C. Y. Detection of Counterfeit Coins Based on Shape and Letterings Features (in press). Proceedings of 28th ISCA International Conference on Computer Applications in Industry and Engineering, San Diego, USA, Oct. 2015.

[9] Reisert, M., Ronneberger, O., Burkhardt, H. A Fast and Reliable Coin Recognition System. Proceedings of 29th DAGM Symposium, vol. 4731, pp. 415-424, Heidelberg, Germany, Sep.12-14, 2007.

[10] Tsai, D.M., Chiang, C.H. Rotation-Invariant Pattern Matching Using Wavelet Decomposition. Pattern Recognition Letters, vol. 23, pp. 191-201. Jan. 2002.

[11] Wei, K.P., He, B., Wang, F., Zhang, T., & Ding Q.J. A Novel Method for Classification of Ancient Coins Based on Image Textures. Proceedings of the Second Workshop on Digital Media and Its Application in Museum & Heritage, pp. 63-66, 2007.

[12] Shen, L., Jia, S., Ji, Z., Chen, W.S. Extracting Local Texture Features for Image-based Coin Recognition. IET Image Processing, vol. 5, pp. 394-401, Aug. 2011.

[13] Huber, R., Ramoser, H., Mayer, K., Penz, H., Rubik, M. Classification of Coins Using An

Eigen-space Approach. Pattern Recognition Letters, vol. 26, pp. 61-75, Jan. 2005.

[14] Nolle, M., Penz, H., Pubik, M., Mayer, K., Hollander, I., Geanec, R. A New Coin Recognition and Sorting System. Proceedings of the 7th International Conference on Digital Image Computing Techniques and Applications, pp. 329-338, Sydney, Australia, Dec. 2003.

[15] Bremananth, R., Balaji, B., Sankari, M., Chitra, A. A New Approach To Coin Recognition Using Neural Pattern Analysis. Proceedings of IEEE INDICON, pp. 366–370, India, Dec. 2005.

[16] Reisert, M., Ronneberger, O., Burkhardt, H. An Efficient Gradient Based Registration Technique for Coin Recognition. Proceedings of the MUSCLE CIS Coin Competition Workshop, pp. 19–31, Berlin, German, Sep. 2006.

[17] Van der Maaten, L. J. P., Postma, E. O. Towards Automatic Coin Classification. Digital Cultural Heritage - Essential for Tourism, Oestereichische Computer Gesellschaft, pp. 19–26, 2006.

[18] Modi, S., Bawa, S. Automated Coin Recognition System using ANN. International Journal of Computer Applications, vol.26, pp. 13-18, Jul. 2011.

[19] Van der Maaten, L. J. P., Boon, P. J. COIN-O-MATIC: A Fast System for Reliable Coin Classification. Proceedings of the MUSCLE CIS Coin Competition Workshop, Berlin, German, pp. 7–17, 2006.

[20] Takacs, G., Chandrasekhar, V., Tsai S.S., Chen, D., Grzeszczuk, R., Girod, B. Fast Computation of Rotation-Invariance Image Features by an Approximate Radial Gradient Transform. IEEE Trans Image Process, vol. 22, pp. 2970-2982. Aug. 2013.

[21] Feng, B.-Y., Sun, K., Atighechian, P., Suen, C.Y. Computer Recognition and Evaluation of Coins, in press, Chen, C.H. (ed.). Handbook of Pattern Recognition and Computer Vision, 5th Edition for World Scientific Publishing Publication, January 2016.

[22] Hough, P.V.C. A Method and Means for Recognizing Complex Patterns, U.S. Patent 3,069,654, Dec. 1962.

[23] Duda, R.O., Hart, P.E. Use of Hough Transformation to Detect Lines and Curves in Pictures. Communications of the ACM, vol. 15, pp. 11-15, Jan. 1972.

[24] Otsu, N. A Thresholding Selection Method from Gray-Level Histograms. IEEE

Transactions on Systems, Man, and Cybernetics, SMC-9, vol. 9, pp. 62-66, Jan. 1979.

[25] Niblack, W. An Introduction to Digital Image Processing. Strandberg Publishing Company, Birkeroed, Denmark, 1985.

[26] Connected components Labeling. http://homepages.inf.ed.ac.uk /rbf/HIPR2 /label.htm, 2003.

[27] Ntirogiannis, K., Gatos, B., Pratikakis, I. A Modified Adaptive Logical Level Binarization Technique for Historical Document Images. Proceedings of the 10th International Conference on Document Analysis and Recognition, IEEE Computer Society, Barcelona, Spain, pp.1171-1175, 2009.

[28] Rafael C.G., Richard E.W. Digital Image Processing, 3rd edition, pp. 628-638, Prentice Hall, 2007.

[29] Jain, A.k. Data Clustering : 50 Years Beyond K-Means. Pattern recognition letters, issue. 31, pp. 651-666, 2010.

[30] Asano, T., Bhattacharya, B., Keil, M., and Yao F. Clustering algorithms based on minimum and maximum spanning trees. Proceedings of the 4th Annual Symposium on Computational Geometry, Urbana Champaign, USA, pp. 252–257, 1988.

[31] Preparata, F., Shamos, M. Computational Geometry: An Introduction. Springer-Verlag, New York, USA, 1985.

[32] Zahn, C.T. Graph-theoretical Methods for Detecting and Describing Gestalt Clusters. IEEE Transactions on Computers, vol. C-20, pp. 68-86, Jan. 1971.

[33] Grygorash, O., Zhou, Y., Jorgensen, Z. Minimum Spanning Tree Based Clustering Algorithms. Proceedings of the 18th International Conference on Tools with Artificial Intelligence, Washington D.C., USA, pp. 73-81, Nov. 2006.

[34] Cormen, T.H., Leiserson, C., Rivest, R.L., Stein, C. Introduction to Algorithms, 3rd edition. The MIT press, Cambridge Massachusetts, USA, 2009.

[35] Kruskal, J. On the Shortest Spanning Subtree and the Traveling Salesman Problem. Proceedings of the American Mathematical Society, vol. 7, pp. 48–50, 1956.

[36] Xu, Y., Olman, V. and Xu, D. Minimum spanning trees for gene expression data clustering. Genome Informatics, vol. 12, pp. 24–33, 2001.

[37] Loncaric, S. A Survey of Shape Analysis Techniques. Pattern Recognition, vol. 31, pp. 983-

1001, 1998.

[38] Materka, A., Strzelecki, M. Texture Analysis Methods – A Review. Institute of Electronics, Lodz, Poland, 1998.

[39] Tuytelaars, T., Mikolajczyk, K. Local Invariant Feature Detectors: A Survey. Foundations and Trends in Computer Graphics and Vision, vol. 3, pp. 177-280, Jan. 2008.

[40] D. Chetverikov and J. Matas. Periodic Textures as Distinguished Regions for Wide-Baseline Stereo Correspondence. In Proceedings of the 2nd International Workshop on Texture Analysis and Synthesis, Copenhagen, Denmark, pp. 25–30, 2002.

[41] Dorko, G., Schmid, C. Selection of Scale Invariant Neighborhoods for Object Class Recognition. Proceedings of the 9th International Conference on Computer Vision, Nice, France, pp. 634–640, Oct. 2003.

[42] Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L. A Comparison of Affine Region Detectors. International Journal of Computer Vision, vol. 65, pp. 43–72, 2005.

[43] Forssen, P.-E., Lowe, D.G. Shape Descriptors for Maximally Stable Extremal Regions. Proceedings of IEEE 11th International Conference on Computer Vision, Rio de Janeiro, Brazil, pp. 1-8, Oct. 2007.

[44] Matas, J., Chum, O., Urban, M., Pajdla, T. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. Image and Vision Computing, vol. 22, pp. 761-767, Sep. 2004.

[45] S. Obdrzalek, Matas, J. Object Recognition Using Local Affine Frames on Maximally Stable Extremal Regions. Toward Category-Level Object Recognition, vol. 4170, pp. 83-104, 2006.

[46] Forssen, P.-E. Maximally Stable Colour Regions for Recognition and Matching. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, Minnesota, USA, Jun. 2007.

[47] Donoser, M., Bischof, H. Efficient Maximally Stable Extremal Region (MSER) Tracking. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 553-560, New York, USA, Jun. 2006.